Numerische Verfahren zur Lösung von Anfangswertaufgaben und zur Generierung von ersten und zweiten Ableitungen mit Anwendungen bei Optimierungsaufgaben in Chemie und Verfahrenstechnik

Inaugural-Dissertation
zur Erlangung der Doktorwürde
der Naturwissenschaftlich-Mathematischen Gesamtfakultät
der Universität Heidelberg

vorgelegt von
Irene Bauer
aus Landsberg am Lech

1. Gutachter: Prof. Dr. Hans Georg Bock

Interdisziplinäres Zentrum für Wissenschaftliches Rechnen (IWR)
der Universität Heidelberg
1999

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit am Interdisziplinären Zentrum für Wissenschaftliches Rechnen (IWR) der Ruprecht-Karls Universität Heidelberg. Sie wurde vom SFB 359 (Reaktive Strömungen, Diffusion und Transport) in den beiden Projekten "Differentiell-algebraische Gleichungssysteme" und "Parameterschätzung und optimale Versuchsplanung für chemische Reaktionssysteme" gefördert. Zudem wurde die Arbeit im BMBF-Projekt "Optimale Versuchsplanung für nichtlineare Prozesse" gefördert. Ich danke sowohl der Deutschen Forschungsgemeinschaft als auch dem Bundesministerium für Bildung und Forschung (BMBF) für die finanzielle Unterstützung und die Bereitstellung der notwendigen Mittel zur erfolgreichen Durchführung dieser Arbeit.

Besonderer Dank gilt meinen beiden Betreuern Professor Dr. Hans Georg Bock und Dr. Johannes Schlöder für die Unterstützung und Förderung dieser Arbeit. Zahlreiche Gespräche, Hinweise und Diskussionen trugen zum Gelingen der Arbeit in ihrer jetzigen Form bei.

Danken möchte ich auch meinen Kolleginnen und Kollegen Dr. Reinhold von Schwerin und Michael Winckler für Hinweise zur Unstetigkeitenbehandlung bei DAE-Systemen, Dr. Marianne von Schwerin und Dr. Johanna Gallitzendörfer, daß sie mir Ihre Codes zur Parameterschätzung für Mehrfachexperimente bei ODE-Systemen zur Verfügung gestellt haben, Dr. Philipp Rosenau für die Mithilfe bei der Implementierung von Sparse-Solvern in DAESOL, Dr. Daniel Leineweber für zahlreiche Diskussionen und Hinweise im Zusammenhang mit Sensitivitätenberechnung und für das Modell der Batch-Destillationskolonne. Stefan Körkel danke ich für die erfolgreiche Zusammenarbeit im BMBF-Projekt sowie Gerd Rüttger für die Automatisierung der Ableitungsgenerierung und für zahlreiche Tests.

Danken möchte ich auch den Mitarbeitern der Gruppe von Dr. Anna Schreieck der BASF AG, inbesondere Dr. Alexander Kud, Dr. Ulrich Daiminger von Aventis und Prof. Dr. Andreas Orth und Frank Beyer der Fachhochschule Frankfurt für die produktive Zusammenarbeit und die zahlreichen Diskussionen im Zusammenhang des BMBF-Projekts "Optimale Versuchsplanung für nichtlineare Prozesse". Mein Dank gilt auch Erik Stein der Arbeitsgruppe von Professor Dr. Gilles aus Magdeburg für zahlreiche Hinweise beim Testen von DAESOL.

Ein spezieller Dank geht an Oliver Bösl in seiner Funktion als Systemadministrator, der bei allen computertechnischen Schwierigkeiten sofort mit Rat und Tat zur Seite stand.

Nicht zuletzt möchte ich mich bei all meinen Kolleginnen und Kollegen bedanken, besonders bei denjenigen, die mich (gerade in der letzten Phase) beim Mittagessen und dem

anschließenden traditionellen Kaffee wieder aufgeheitert haben.

Alle in der Arbeit dargestellten Rechnungen wurden auf einer SGI O2 mit einem 180 MHz IP32 R5000 Prozessor durchgeführt. Das verwendete Betriebssystem war IRIX 6.3.

Inhaltsverzeichnis

Einleitung

Die computergestützte Simulation und Optimierung von Prozessen aus Chemie und Verfahrenstechnik hat in den letzten Jahren immer mehr an Bedeutung gewonnen. Einsatzgebiete sind etwa die Maximierung der Ausbeute oder die Minimierung von Abfallprodukten oder des Energieaufwands. Aber schon die reine Simulation ist oft wichtig, um die Verhaltensweise des Prozesses und das Zusammenspiel der oft stark nichtlinearen und teils gekoppelten Mechanismen besser zu verstehen. Dies kann auch bei dem Bau einer neuen Anlage – für deren Prozeß bis dato nur Wissen aus dem Labormaßstab vorliegt – von großem Nutzen sein. Wichtig sowohl für die Simulation als auch für die Optimierung ist dabei, daß das zugrundeliegende Modell den realen Prozeß sehr gut widerspiegelt.

Die Modellierung von chemischen und verfahrenstechnischen Prozessen führt oft auf ein System von differentiell-algebraischen Gleichungen (engl. *differential algebraic equations*, kurz DAE). Die DAE-Systeme sind meist steif, nichtlinear und können sehr groß sein. Die Modelle enthalten zudem oft Parameter, die nur unzureichend bekannt sind und nicht direkt gemessen werden können. Dabei handelt es sich oft um spezielle Größen, die die Kinetik oder verfahrenstechnische Komponenten des Prozesses beschreiben.

In der vorliegenden Arbeit werden effiziente Methoden zur numerischen Behandlung von DAE-Systemen mit Anwendungen in Chemie und Verfahrenstechnik vorgestellt. Wir betrachten die Lösung von Anfangswertproblemen bei DAE-Systemen und von Optimierungsproblemen in Parameterschätzung und Versuchsplanung. Insbesondere beschreiben wir effiziente Implementierungen zur Auswertung der für die Optimierung notwendigen Ableitungen auf Basis von Techniken der Internen Numerischen Differentiation (kurz IND), deren Idee und erste Realisierungen auf Bock [Boc81] zurückgehen.

Numerische Lösung von Anfangswertproblemen bei DAE-Systemen

Das im Rahmen dieser Arbeit weiterentwickelte Programmpaket DAESOL baut auf den Arbeiten von Bleser [Ble86] und Eich [Eic87] auf. Es ist ein BDF-Verfahren (engl. *backward differentiation formulae*) variabler Ordnung und Schrittweite, das über die letzten 10 bis 15 Jahre in der Arbeitsgruppe von Bock entwickelt wurde. Es löst die allgemeine Klasse von quasilinearen impliziten DAE-Systemen vom Index 1. Je nach Größe des zu behandelnden Systems kann eine Dense- oder Sparse-Version mit jeweils unterschiedlichen Lineare-Algebra-Lösern verwendet werden.

BDF-Verfahren für gewöhnliche Differentialgleichungen (engl. *ordinary differential equations*, kurz ODE) gehen zurück auf Curtiss und Hirschfelder [CH52]. Gear übertrug die Methoden 1971 [Gea71] erstmals auf DAE-Systeme. Seitdem wurden mehrere Codes zur Integration von ODE-und DAE-Systemen basierend auf BDF-Formeln entwickelt. Der bekannteste hiervon ist DASSL von Petzold [Pet82a, Pet91, BCP96] und seine Weiterentwicklung DASPK von Li und Petzold [LP99a, LP99b].

Nahezu alle Integratoren verfügen über eine adaptive Ordnungs- und Schrittweitensteuerung. Sie unterscheiden sich jedoch hinsichtlich der Darstellung und Abspeicherung des Interpolationspolynoms und der Schätzung des Fehlers. Die Fehlerschätzung berücksichtigt mittlerweile bei fast allen Mehrschrittverfahren das tatsächliche nichtäquidistante Gitter, jedoch beruhen die Approximationen für die Schrittweitensteuerung meist (so zum Beispiel in DASSL/DASPK und in dem Code VODE von Brown et al. [BBH89, BHB98]) auf Formeln auf äquidistantem Gitter.

Der Hauptrechenaufwand bei der numerischen Behandlung von steifen DAE-Systemen liegt in der Regel im Lösen der bei impliziten Verfahren auftretenden nichtlinearen Gleichungssysteme. Das Nullstellenproblem wird fast immer mit einem vereinfachten Newton-Verfahren gelöst. Die Jacobi-Matrix hängt dabei zum einen von der Schrittweite und zum anderen von Ableitungen der Modellfunktionen des DAE-Systems ab. Alle Integratoren verfügen über Strategien, die einen möglichst geringen Aufwand für die Auswertung und Zerlegung der Jacobi-Matrix anstreben; die zerlegte Matrix wird so lange wie möglich eingefroren. Bei schlechter Konvergenz des Newton-Verfahrens wird in den meisten Fällen die gesamte Jacobi-Matrix neu berechnet und zerlegt. Lediglich DAESOL und in ähnlicher Weise VODE [BBH89, BHB98] verfügen über eine sogenannte Monitor-Strategie. Es wird berücksichtigt, daß eine Änderung der Schrittweite oft schon zu Konvergenzproblemen führen kann. In diesen Fällen wird zunächst nur die Jacobi-Matrix neu zerlegt, die Ableitungen der Modellfunktionen aber konstant gehalten. Die Auswertung und Zerlegung der Jacobi-Matrix des vereinfachten Newton-Verfahrens wird in DAESOL nach dem lokalen Kontraktionssatz von Bock [Boc87] adaptiv gesteuert. Die Monitor-Strategie reduziert den Aufwand der Lineare-Algebra-Teilprobleme und führt in den meisten Fällen insbesondere zu einer deutlich geringeren Anzahl an Auswertungen der Ableitungen der Modellfunktionen.

Da man in letzter Zeit immer mehr dazu übergeht, nicht nur die einzelnen Komponenten einer Anlage zu modellieren, sondern gerade am Verhalten des gesamten Prozesses und der Wechselwirkung der einzelnen Komponenten interessiert ist, werden die dabei auftretenden Systeme oft sehr groß und komplex. Falls das Modell gleichzeitig stark nichtlinear in den algebraischen Gleichungen ist und die Anfangswerte für einige der algebraischen Variablen nicht bekannt sind, so stellt oft schon die konsistente Initialisierung die erste Schwierigkeit dar. Ein Vollschritt-Newton-Verfahren, wie es in den meisten Integratoren zur Lösung des Nullstellenproblems implementiert ist, hat bei stark nichtlinearen Gleichungen nur einen sehr kleinen lokalen Konvergenzbereich und konvergiert bei ungenauen Schätzungen für die Anfangswerte oft nicht mehr.

DASPK [LP99a, LP99b] stellt zur konsistenten Initialisierung ein gedämpftes Newton-

Verfahren bereit. Bei stark nichtlinearen Gleichungen konvergiert das Verfahren allerdings trotz Dämpfungsfaktor oft nicht mehr. In DAESOL wurde ein Homotopie-Verfahren variabler Schrittweite implementiert. Die Schrittweite des Homotopie-Verfahrens und der Aufwand für die Lineare-Algebra-Teilprobleme zur Lösung der nichtlinearen Gleichungssyteme mit Hilfe eines vereinfachten Newton-Verfahrens wird adaptiv nach dem lokalen Kontraktionssatz für Homotopie-Verfahren (siehe Bock [Boc87] für den Kontraktionssatz für Homotopie-Verfahren bei Parameterschätzproblemen) gesteuert. Der Startwert für den aktuellen Schritt wird mit Hilfe einer Extrapolation ähnlich zum BDF-Verfahren gewonnen. Der Benutzer kann Standard-Homotopien auswählen oder selbst eine physikalisch begründete Homotopie bereitstellen.

Zur Lösung der steifen DAE-Systeme haben sich BDF-Verfahren im allgemeinen sehr gut bewährt. Enthält das Modell allerdings viele Unstetigkeiten, so verlieren die Mehrschrittverfahren einiges an Effizienz, da dann der Mehraufwand in der Startphase einen erheblichen Anteil an der Gesamtrechenzeit ausmachen kann. Auf Gear [Gea80] geht die Idee zurück, ein Einschrittverfahren höherer Ordnung zur Generierung der für Mehrschrittverfahren benötigten zurückliegenden Werte zu verwenden. Gear und Brankin et al. [BGS88] verwendeten ein explizites Runge-Kutta-Verfahren zur Generierung der zurückliegenden Werte sowohl für explizite als auch implizite Mehrschrittverfahren. Von Schwerin und Bock [vSB95, vS97] entwickelten ein explizites Runge-Kutta-Verfahren zum Neustart eines Adams-Verfahrens. Dabei sind einige der internen Stufen des Runge-Kutta-Verfahrens von höherer Ordnung, so daß ein Schritt des Einschrittverfahrens ausreicht, um alle zurückliegenden Werte für die anschließende Integration mit dem Adams-Verfahren zu erhalten. Diese Idee wurde auch für den Runge-Kutta-Starter für das BDF-Verfahren in DAESOL übernommen und ein implizites Runge-Kutta-Verfahren konstruiert, das in einem Schritt alle zurückliegenden Werte bereitstellt. Die Lösung der impliziten Gleichungssysteme und das für die Fehlerschätzung eingebettete Verfahren sind dabei genau auf die Vorgehensweise im BDF-Verfahren zugeschnitten.

Weitere Integratoren zur Lösung von steifen DAE-Systemen sind etwa die von Hairer und Wanner entwickelten impliziten Runge-Kutta-Verfahren RADAU [HW98], RADAU5 [HW96a] und SDIRK4 [HW96b] und das Extrapolationsverfahren LIMEX von Deuflhard et al. [DHZ87] und seine Weiterentwicklung von Ehrig und Nowak [EN99].

Optimierungsprobleme bei Parameterschätzung und Versuchsplanung

Um zuverlässige Aussagen über das Verhalten eines Prozesses treffen zu können, muß das mathematische Modell den Prozeß genau beschreiben. Dies wird schon für die Simulation benötigt, aber erst recht natürlich für die Optimierung des Prozesses hinsichtlich unterschiedlicher Gütefunktionen, etwa der Maximierung der Produktqualität oder der Reduzierung des Energieaufwands. Auch für Untersuchungen zur Prozeßsicherheit sind validierte Modelle nötig.

Zur Schätzung der im Modell auftretenden unbekannten Parameter müssen Meßdaten erhoben werden. Wir betrachten Optimierungsprobleme zur Parameterschätzung, die auf

ein Least-Squares-Funktional zur Minimierung des Abstands zwischen Meßdaten und Modellantwort (gewichtet mit der Standardabweichung des Meßfehlers) führen. Zur Lösung verwenden wir den Mehrzielansatz, der erstmals von Bock [Boc78] auf die Klasse der Mehrpunkt-Randwertprobleme angewendet wurde. Wir erhalten ein nichtlineares endlichdimensionales beschränktes Optimierungsproblem, das wir mit einem verallgemeinerten Gauß-Newton-Verfahren lösen (siehe Bock [Boc87], Schlöder [Sch88] und von Schwerin [vS98]).

Die für die Parameterschätzung notwendigen Experimente können dabei sehr teuer sein. Zudem können unterschiedliche Experimente und Messungen sehr viel oder auch nur sehr wenig Information für die zu schätzenden Parameter liefern. Bei komplexen Prozessen kann auch ein erfahrener Experimentator mit einem intuitiv aufgestellten Versuchsplan oft nicht mehr alle Wechselwirkungen erfassen. Zur Reduzierung der Experimentalkosten und zur Maximierung der statistischen Güte der zu schätzenden Parameter verwenden wir daher Methoden der optimalen Versuchsplanung.

Für nichtlineare Regressionsmodelle wurde die optimale Versuchsplanung bereits Ende der 50er Jahre untersucht, etwa von Box und Lucas [BL59] oder von Kiefer und Wolfowitz [KW59]. Es wird eine Funktion (meist die Determinante) auf der (näherungsweisen) Kovarianz- oder Informationsmatrix minimiert bzw. maximiert. Freie Variablen sind die das Experiment beschreibenden Größen. Für die dort betrachteten Beispiele wurde die Approximation der Kovarianzmatrix und ihrer Ableitungen analytisch dargestellt. Dieser Ansatz wird auch heute noch teilweise übernommen, siehe etwa Dovíc et al. [DRAD94] oder Rudolph und Herrendörfer [RH95].

In der vorliegenden Arbeit betrachten wir Versuchsplanungsprobleme, die auf den Ansatz für nichtlineare Regressionsmodelle mit den dabei verwendeten klassischen Gütekriterien zurückgehen, Kern des zugrundeliegenden nichtlinearen Modells ist hier allerdings ein DAE-System. Die mathematische Formulierung des Versuchsplanungs-Optimierungsproblems führt auf ein Optimal-Steuerungsproblem mit einem sehr komplexen Zielfunktional.

Versuchsplanungsprobleme für Prozesse, deren Modellierung auf ein ODE-System führt, wurden von Lohmann et al. [LBS92, Loh93] untersucht. Die Optimierung sucht die besten aus den möglichen Messungen aus, die Auslegung der Experimente wird allerdings nicht mitoptimiert. Die 0-1-Bedingungen zur Auswahl der Messungen werden dabei relaxiert. In [Loh93] beschreibt Lohmann eine geschickte Erzeugung der Ableitungen des Zielfunktionals und insbesondere der Kovarianzmatrix nach den Gewichten an die Messungen.

Hilf [Hil96] untersuchte Versuchsplanungsprobleme für unbeschränkte Parameterschätzprobleme in der Mechnaik. Das Zielfunktional ist eine Gütefunktion auf der Informationsmatrix.

In unserem Ansatz zur Lösung der Optimal-Steuerungsprobleme in der Versuchsplanung minimieren wir eine Gütefunktion auf der Kovarianzmatrix. Dies erlaubt auch die Versuchsplanung für beschränkte Parameterschätzprobleme. Sowohl die Gewichte an die Messungen als auch die Steuergrößen und Steuerfunktionen, die die Fahrweise des Prozesses

beschreiben, können optimiert werden. Für die Auswahl der Gewichte an die Messungen verwenden wir eine relaxierte Formulierung. Die unendlich-dimensionale Lösungstrajektorie des DAE-Systems und die Steuerfunktionen werden parametrisiert. Dies führt auf ein nichtlineares endlich-dimensionales Optimierungsproblem mit einem Zielfunktional, das implizit von ersten Ableitungen der Lösungstrajektorie des DAE-Systems nach Anfangswerten und Parametern abhängt.

Wir lösen das Optimierungsproblem mit einem SQP-Verfahren. Dies erfordert nicht nur die Bereitstellung der Lösung des DAE-Systems, sondern insbesondere auch der ersten und zweiten gemischten Ableitungen der Lösungstrajektorie nach Anfangswerten, Parametern und Steuergrößen.

Der von uns betrachtete Ansatz erlaubt dabei nicht nur die Optimierung der statistischen Güte der zu schätzenden Parameter, sondern auch von abgeleiteten Größen, an denen der Benutzer speziell interessiert ist, etwa für eine anschließende Optimierung.

Generierung von Ableitungen der Löung des DAE-Systems

Zusätzlich zur Simulation des DAE-Systems müssen für das Optimierungsproblem zur Parameterschätzung (wie auch für eine anschließende Optimierung des Prozesses) erste Ableitungen der Lösung des DAE-Systems nach Anfangswerten und Parametern (bzw. Anfangswerten und Steuergrößen) mit der von der Optimierung geforderten Genauigkeit bereitgestellt werden. Für das Versuchsplanungs-Optimierungsproblem sind zusätzlich zweite gemischte Ableitungen der Lösungstrajektorie nach Anfangswerten, Parametern und Steuergrößen nötig. Die Leistungsfähigkeit der Optimierungsverfahren hängt dabei sehr stark von der effizienten Lösung der DAE-Systeme und der Generierung der entsprechenden Ableitungen ab.

Das Programmpaket DDASAC von Caracotsios und Stewart [CS85], eine Erweiterung des Codes DASSL von Petzold [Pet82a], berechnet Ableitungen der Lösungstrajektorie mit Hilfe der Variationsdifferentialgleichung. Dabei wird das lineare Gleichungssystem immer direkt gelöst, auch wenn die auftretenden Systeme sehr groß sind.

Maly und Petzold [MP96] implementierten eine Weiterentwicklung von DASSL zur Ableitungsgenerierung, DASSLSO, die ein Gesamtsystem aus Nominaltrajektorie und Sensitivitätsgleichungen aufstellt und das vereinfachte Newton-Verfahren auf das volle System anwendet. Für das Newton-Verfahren wird eine Approximation der Jacobi-Matrix verwendet, so daß für die Auswertung und Zerlegung trotzdem nur eine Matrix von der Dimension der Zustandsvariablen (und nicht des Gesamtsystems) benötigt wird.

Das von Li und Petzold [LP99a, LP99b] entwickelte Programmpaket DASPK stellt zum einen die Variante von DASSLSO zur Verfügung, zudem zwei weitere Möglichkeiten: das direkte Lösen der linearen Gleichungssysteme der diskretisierten Sensitivitätsgleichungen und die Anwendung eines vereinfachten Newton-Verfahrens. Letzterer Fall entspricht allerdings nicht den Prinzipien der IND: der Fehler im Newton-Verfahren wird kontrolliert, die berechneten Sensitivitäten sind aber nicht mehr die direkten Ableitungen der mit Hilfe des Diskretisierungsschemas berechneten Nominaltrajektorie.

Die in DAESOL implementierte Ableitungsgenerierung folgt den Prinzipien der IND. Dabei sind in DAESOL unterschiedliche Varianten implementiert, die für unterschiedliche Problemstellungen sinnvoll sind. Sie unterscheiden sich speziell in der Lösung der linearen Gleichungssyteme - direktes Lösen der Gleichungssysteme oder Ableitung auch des Newton-Verfahrens zur Lösung der diskretisierten Systeme der Nominaltrajektorie. In letzterem Fall führt die Anwendung der Prinzipien der IND darauf, daß die Monitor-Strategie aus der Nominaltrajektorie auch auf die Lösung der Sensitivitätsgleichungen angewendet wird.

Während die anderen Integratoren nur erste Ableitungen der Lösung des DAE-Systems berechnen, können mit DAESOL auch zweite Ableitungen generiert werden, wie sie zum Beispiel bei der Lösung der Optimierungsprobleme zur Versuchsplanung benötigt werden. Die Strukturen aufgrund der relaxierten Formulierung, wie sie gerade im Opimierungskontext sinnvoll sind, werden voll ausgenutzt.

Zudem ist DAESOL unseres Wissens der einzige Integrator für steife DAE-Systeme, der die Generierung von Richtungsableitungen direkt unterstützt. Diese treten etwa im reduzierten Ansatz zur Parameterschätzung auf oder bei der Lösung von Optimierungsproblemen mit einem reduzierten SQP-Verfahren wie sie bei Leineweber [Lei99] beschrieben sind. DAESOL wird hier mit großem Erfolg eingesetzt.

Gliederung der Arbeit

Die vorliegende Arbeit ist wie folgt aufgebaut. In Kapitel 1 untersuchen wir zunächst die theoretischen Grundlagen zur Lösung von Anfangswertproblemen bei DAE-Systemen, insbesondere die Definition des Index sowie Existenz und Eindeutigkeit von Lösungen.

Kapitel 2 befaßt sich mit Konsistenz- und Konvergenzaussagen von Mehrschrittverfahren, insbesondere von BDF-Verfahren. Wir betrachten zunächst die Aussagen für äquidistante Gitter und konstante Ordnung und untersuchen, inwieweit sich diese auf Verfahren variabler Schrittweite und Ordnung übertragen lassen. Im Anschluß geben wir die speziellen Implementierungen in DAESOL an wie Fehlerschätzung und Schrittweiten- und Ordnungssteuerung und die Monitor-Strategie zur Lösung der nichtlinearen Gleichungssysteme des impliziten Verfahrens.

Für eine effiziente Behandlung der Startphase wurden in DAESOL spezielle Strategien entwickelt und implementiert, auf die wir in Kapitel 3 eingehen. Zur konsistenten Initialisierung stehen ein Vollschritt-Newton-Verfahren und ein Homotopie-Verfahren zur Verfügung. Im Optimierungskontext wird eine relaxierte Formulierung der algebraischen Gleichungen angeraten und in DAESOL unterstützt. Für den Neustart des BDF-Verfahrens wurde ein speziell auf das Verfahren zugeschnittenes implizites Runge-Kutta-Verfahren konstruiert und implementiert.

Kapitel 4 widmet sich dem Optimierungsproblem in der Parameterschätzung. Wir zeigen Lösungsansätze für das verallgemeinerte Gauß-Newton-Verfahren für das beschränkte Optimierungsproblem und untersuchen lokale Konvergenzaussagen. Für separable Nebenbedingungen kann ein reduzierter Ansatz verwendet werden. Werden die Parameter aus

Meßdaten aus mehreren Experimenten geschätzt, so werden auch die dabei auftretenden Strukturen ausgenutzt. Schließlich geben wir eine Berechnungsvorschrift für eine Approximation der Kovarianzmatrix der geschätzten Größen an.

Die Parameterschätzung ist das unterliegende Problem für das in Kapitel 5 dargestellte Optimal-Steuerungsproblem zur Versuchsplanung. Basierend auf der im letzten Kapitel aufgestellten Kovarianzmatrix formulieren wir das Versuchsplanungs-Optimierungsproblem mit seinen Nebenbedingungen (Innere-Punkt-Bedingungen, Steuerungsbeschränkungen, Zustandsbeschränkungen, etc.). Wir entwickeln einen direkten Ansatz zur Lösung des Optimal-Steuerungsproblems und wenden ein strukturiertes SQP-Verfahren auf das resultierende endlich-dimensionale Optimierungsproblem an. Inbesondere zeigen wir eine geschickte Berechnung der für die Optimierung notwendigen Ableitungen, die aus dem Zusammenspiel von semi-analytischen Ableitungen, Automatischer Differentiation und Techniken der Internen Numerischen Differentiation zur Berechnung der Ableitungen der Lösung des DAE-Systems besteht.

Die effiziente Generierung der für die Optimierungsprobleme zur Parameterschätzung und Versuchsplanung benötigten ersten und zweiten gemischten Ableitungen der Lösungstrajektorie des DAE-Systems mit Techniken der IND leiten wir in Kapitel 6 her. Im speziellen diskutieren wir die unterschiedlichen Varianten, die in DAESOL implementiert sind.

In Kapitel 7 werden einige Beispiele aus Chemie und Verfahrenstechnik diskutiert. Wir beginnen mit einer Batch-Destillationskolonne, die zahlreiche stark nichtlineare algebraische Gleichungen enthält, und zeigen die effiziente Berechnung konsistenter Anfangswerte in DAESOL. Im Anschluß vergleichen wir den Integrator DAESOL für einige repräsentative Testbeispiele mit anderen Integratoren sowohl für die Lösung von Anfangswertproblemen als auch für die Ableitungsgenerierung. Anhand zweier chemischer Reaktionen – Phosphin- und Urethan-Reaktion in einem Semi-Batch-Reaktor – zeigen wir die effiziente numerische Lösung der Versuchsplanungs-Optimierungsprobleme, insbesondere die Effizienz einer sequentiellen Vorgehensweise aus Parameterschätzung und Versuchsplanung.

In Kapitel 8 werden die grundlegenden Methoden und Neuerungen nochmals zusammengefaßt und ihre Leistungsfähigkeit gerade im Hinblick auf die numerischen Ergebnisse diskutiert. Es werden offene und weiterführende Fragen diskutiert.

Kapitel 1

Theoretische Grundlagen zur Behandlung von Anfangswertproblemen bei DAE-Systemen

Die Modellierung chemischer Prozesse führt in der Regel auf ein System von Differentialgleichungen, teilweise gekoppelt mit algebraischen Gleichungen, sogenannten DAEs (engl. differential algebraic equations). Die Differentialgleichungen treten z. B. bei der Modellierung von chemischen Reaktionen oder Transportgleichungen auf, algebraische Gleichungen rühren z. B. aus Erhaltungsgleichungen oder Steady-state-Annahmen. Auch die räumliche Diskretisierung von partiellen Differentialgleichungen mit der Linienmethode führt häufig auf DAEs.

DAEs unterscheiden sich von gewöhnlichen Differentialgleichungen (engl. ordinary differential equations, kurz ODEs) dahingehend, daß die Lösungstrajektorie auf der Mannigfaltigkeit liegen muß, die von den algebraischen Gleichungen aufgespannt wird. Dadurch ergeben sich Unterschiede bei der Frage nach Existenz und Eindeutigkeit von Lösungen und bei der numerischen Behandlung der DAE-Systeme. Bei der numerischen Lösung von DAEs ist der sogenannte Index eine wichtige Kenngröße.

1.1 Index einer DAE

Betrachte die semi-explizite DAE

$$\dot{y} = f(t, y, z) \tag{1.1a}$$
$$0 = g(t, y, z). \tag{1.1b}$$

Dabei bezeichne $t \in \mathbb{R}$ die Zeit beziehungsweise den Ort, $y(t) \in \mathbb{R}^{n_y}$ die differentiellen und $z(t) \in \mathbb{R}^{n_z}$ die algebraischen Variablen mit $n_y, n_z \geq 0$. Mit $f \in \mathcal{C}(\mathbb{R}^{n_y})$ und $g \in \mathcal{C}(\mathbb{R}^{n_z})$

seien die differentiellen beziehungsweise algebraischen Gleichungen beschrieben. Totale zeitliche Differentiation der algebraischen Gleichungen ergibt

$$g_t + g_y\,\dot{y} + g_z\,\dot{z} = 0 \tag{1.1b$'$}$$

Ist g_z regulär, so kann man Gleichung (1.1b$'$) nach $\dot{z}$ auflösen

$$\dot{z} = -g_z^{-1}\,(g_t + g_y\,\dot{y})$$

und erhält ein System von gewöhnlichen Differentialgleichungen (1.1a, 1.1b$'$) in y und z. Die DAE (1.1a, 1.1b) heißt dann vom (differentiellen) Index 1, da sie mit einer zeitlichen Differentiation in eine ODE überführt werden kann.

Ist g_z singulär, so bringt man (1.1b$'$) wieder in die Form von (1.1a, 1.1b) – mit neuen $\tilde{y}$ und $\tilde{z}$ und eventuell einer Substitution von Zeilen aus (1.1a) in (1.1b$'$) – und leitet die so gewonnenen algebraischen Gleichungen $\tilde{g}$ wieder nach der Zeit ab. Ist $\tilde{g}_{\tilde{z}}$ regulär, kann $\tilde{z}$ aus den entstandenen Gleichungen bestimmt werden. Dieser Prozeß wird so lange fortgeführt, bis man ein System von Differentialgleichungen für alle z-Komponenten erhält.

Der Begriff des differentiellen Index eines DAE-Systems wurde von Gear [Gea88] eingeführt und ist wie folgt definiert:

Definition 1.1.1 (Differentieller Index eines DAE-Systems)
Die nichtlineare implizite DAE

$$F(t, x, \dot{x}) = 0 \tag{1.2}$$

heißt vom (differentiellen) Index k*, falls k die kleinste Zahl ist, so daß $\dot{x}$ eindeutig bestimmt ist durch die $k+1$ Gleichungen*

$$\begin{aligned} F(t, x, \dot{x}) &= 0 \\ \frac{d}{dt}F(t, x, \dot{x}) &= 0 \\ &\vdots \\ \frac{d^k}{dt^k}F(t, x, \dot{x}) &= 0\,. \end{aligned}$$

Der differentielle Index gibt an, wieviele Differentiationen nach der unabhängigen Variable t notwendig sind, um das System in eine gewöhnliche Differentialgleichung zu überführen. Damit werden die algebraischen Gleichungen des DAE-Systems charakterisiert.

Bemerkung 1.1.1 (DAEs vom Index größer als 1)
Eine DAE vom Index 2 oder höher kann durch totale zeitliche Differentiation der algebraischen Gleichungen ((k-1)-mal für Index k) in eine DAE vom Index 1 transformiert werden. Die analytische Lösung des Anfangswertproblems für das indexreduzierte System erfüllt die ursprünglichen algebraischen Gleichungen und deren erste bis (k-1)-te Ableitungen,

stimmt also mit der analytischen Lösung des Anfangswertproblems für die ursprüngliche DAE überein. Die algebraischen Gleichungen zusammen mit den ersten k-1 Ableitungen nennt man somit Invarianten des indexreduzierten Systems. Probleme entstehen bei der numerischen Lösung des Anfangswertproblems für das indexreduzierte System: Diskretisierungs- und Rundungsfehler verursachen eine Drift weg von der Mannigfaltigkeit, die durch die Invarianten aufgespannt wird. Um diese Drift bei der numerischen Lösung des Anfangswertproblems für das indexreduzierte DAE-System zu vermeiden, behilft man sich meist mit einer Projektion der Lösungstrajektorie auf die Invarianten wie es zum Beipiel in den Integratoren von Eich [Eic92, Eic93], Petzold [BCP96] und von Schwerin [vS97] implementiert ist. Einen Überblick über Verfahren, die Projektionsmethoden anwenden, gibt Eich in [Eic93]. Für Systeme aus der Mechanik, die in der Regel vom Index 3 sind, werden auch oft Regularisierungstechniken wie zum Beispiel eine Stabilisierung nach Baumgarte [Bau72] oder daraus abgeleitete Methoden eingesetzt (siehe zum Beispiel den Artikel von Petzold et al. [PRM97]).

Auf DAEs vom Index 2 wird häufig direkt – wie bei der Lösung von DAEs vom Index 1 beziehungsweise von ODEs – die Diskretisierung angewendet. Arnold [Arn91] und Tischendorf [Tis95] geben Fehlerschätzungen für die direkte Lösung von quasilinearen DAEs vom Index 2 der Form

$$A(t)\,\dot{x}(t) + g(t, x(t)) = 0,$$

bei denen der Nullraum von $A(t)$ konstant ist, mit BDF-Vefahren an. Petzold und Lötstedt [Pet82b, PL86] schlagen vor, bei bestimmten DAEs vom Index 2 (im Falle semi-expliziter DAEs der Form (1.1) bedeutet dies, daß $\partial g/\partial y \equiv 0$) die Fehlerschätzung nur auf dem Fehler in den differentiellen Variablen zu basieren, da der Fehler in den algebraischen Variablen an früheren Punkten nicht direkt in die aktuelle Fehlerschätzung eingeht (siehe zum Beispiel die Approximation des globalen Fehlers bei BDF-Verfahren in (2.18)). Ein Nachteil dabei ist allerdings, daß die Auswertung der algebraischen Variablen nicht mehr fehlerkontrolliert ist. Falls Werte der algebraischen Variablen zwischen den Gitterpunkten benötigt werden, so kann dies zu Genauigkeitsverlusten bei der Auswertung an den interpolierten Punkten führen.

Bemerkung 1.1.2 (Automatische Bestimmung des Index und der Invarianten) *Pantelides entwickelte [Pan88] einen Algorithmus zur Bestimmung des sogenannten strukturellen Index eines DAE-Systems. Das Verfahren beruht auf einer Analyse der Struktur der bei der Differentiation auftretenden Systeme. Der differentielle Index, der von den aktuellen Werten in t und x abhängt, die sich entlang der Lösung ändern, und zunächst nicht bekannt sind, ist kleiner gleich dem strukturellen Index, kann aber mit diesem Verfahren nicht bestimmt werden.*

In letzter Zeit wurden deshalb Algorithmen entwickelt, die darüberhinaus die bei der Differentiation auftretenden Systeme numerisch analysieren. Bachmann et al. [BBMP90] bestimmen zunächst die rein algebraischen Gleichungen, differenzieren diese und benutzen die differentiellen Gleichungen, um alle auftretenden Zeitableitungen der Zustandsvariablen in den abgeleiteten algebraischen Gleichungen zu eliminieren. Von den differentiellen

Gleichungen, die zur Elimination beigetragen haben, werden genau so viele entfernt, wie durch die Differentiation neu hinzugekommen sind. Dadurch bleibt das System wohldefiniert.

Der Algorithmus von Mattsson und Söderlind [MS93] ist ähnlich, jedoch werden nicht differentielle Gleichungen eliminiert, sondern pro hinzugefügter algebraischer Gleichung wird eine algebraische Variable eingeführt, die eine zeitliche Ableitung der Zustandsvariablen ersetzt. Die zusätzlich eingeführten Variablen werden „dummy derivatives" genannt.

Das Verfahren von Pantelides et al. [PSV94] ist ähnlich zu dem von Mattsson und Söderlind, basiert allerdings nur auf dem von Pantelides in [Pan88] entwickelten Algorithmus, automatischer Differentiation zur Berechnung der Ableitungen der algebraischen Gleichungen und numerischer Analyse und ist somit voll automatisch. Die Berechnung konsistenter Anfangswerte geschieht mit Hilfe des erweiterten Systems, für die Integration wird nur das auf Index 1 reduzierte System verwendet und auf die Invarianten projiziert.

1.2 Störungsindex einer DAE

Ein zweites wichtiges Kriterium bei der numerischen Behandlung von DAEs ist der Störungsindex, der von Hairer et al. [HLR89] eingeführt wurde. Er dient als ein Maß für die Anfälligkeit des Systems auf kleine Störungen in der rechten Seite oder in den Anfangswerten.

Definition 1.2.1 (Störungsindex eines DAE-Systems)
Für eine nichtlineare implizite DAE $F(t, x, \dot{x}) = 0$ ist der Störungsindex *entlang einer Lösungstrajektorie $x(t)$, $t \in [t_0, t_f]$, definiert als die kleinste Zahl k, so daß für alle Lösungstrajektorien $\hat{x}$ des gestörten Systems $F(t, \hat{x}, \dot{\hat{x}}) = \delta(t)$ eine Ungleichung*

$$\begin{aligned}\|\hat{x}(t) - x(t)\| \leq C \Bigg(& \|\hat{x}(0) - x(0)\| + \max_{0\leq\tilde{t}\leq t} \| \int_0^{\tilde{t}} \delta(\tau) d\tau \| \\ & + \max_{0\leq\tilde{t}\leq t} \|\delta(\tilde{t})\| + \ldots + \max_{0\leq\tilde{t}\leq t} \|\delta^{(k-1)}(\tilde{t})\| \Bigg)\end{aligned} \tag{1.3}$$

existiert, falls $\delta(t)$ hinreichend klein ist.

Die Konstante C hängt von der Funktion F und von der Länge des Intervalls $[t_0, t_f]$ ab.

Der Störungsindex ist 0, wenn eine Abschätzung der Form

$$\|\hat{x}(t) - x(t)\| \leq C \left(\|\hat{x}(0) - x(0)\| + \max_{0\leq\tilde{t}\leq t} \| \int_0^{\tilde{t}} \delta(\tau) d\tau \| \right)$$

existiert.

Der Störungsindex zeigt den Einfluß von Rundungsfehlern in F. Ist die DAE vom Störungsindex k, so tritt in Gleichung (1.3) die (k-1)-te Ableitung der Störung δ auf

der rechten Seite der Ungleichung auf. Auch wenn die Störung selbst ziemlich klein ist, kann deren Ableitung große Werte annehmen. Für die numerische Lösung heißt dies, daß Diskretisierungs- und Rundungsfehler des Integrators mit einer Größenordnung von $\mathcal{O}(\frac{1}{h^{k-1}})$ in die Lösung eingehen, wenn h die maximale Schrittweite während des Integrationsverlaufs ist. Dies führt zu numerischen Problemen für Systeme mit Störungsindex größer oder gleich 2.

Für ODEs mit Lipschitz-stetiger rechter Seite ist der Störungsindex gleich 0.

Für semi-explizite DAEs ist der differentielle Index gleich dem Störungsindex. Dies gilt nicht allgemein für nichtlineare implizite DAEs, wie das folgende Beispiel von Hairer et al. [HLR89] zeigt:

Beispiel 1.1
Gegeben sei das DAE-System

$$\begin{aligned} \dot{x}_1 - x_3\,\dot{x}_2 + x_2\,\dot{x}_3 &= 0 \\ x_2 &= 0 \\ x_3 &= 0\,. \end{aligned}$$

Für die Anfangswerte $x(0) = (0,0,0)^T$ *ist die Lösung* $x \equiv (0,0,0)^T$.

Nimmt man eine kleine Störung $\delta(t) = (0, \varepsilon sin\omega t, \varepsilon cos\omega t)^T$, ε *klein, auf der rechten Seite des DAE-Systems an, so erhält man für das gestörte DAE-System die Lösung* $\hat{x} = (\varepsilon^2\omega t, \varepsilon sin\omega t, \varepsilon cos\omega t)^T$. *Für festes* ε, ε *klein, und* $\omega \to \infty$ *kann man in (1.3) keine Abschätzung mit* $k = 1$ *finden, sondern erst wenn auch die Terme für* $k = 2$, *also* $\max|\delta(t)|$, *auf der rechten Seite von (1.3) mitberücksichtigt werden. Das System hat somit Störungsindex 2 obwohl der differentielle Index gleich 1 ist.*

1.3 Konsistente Initialisierung

Ein Problem bei der numerischen Lösung von DAEs besteht in der Bestimmung konsistenter Anfangswerte. Für eine nichtlineare implizite DAE muß

$$F(t_0, x_0, \dot{x}_0) = 0 \tag{1.4a}$$

gelten. Falls das System den Index $k \geq 2$ hat, so müssen die Anfangswerte sowohl die ursprüngliche DAE als auch die abgeleiteten Gleichungen

$$\begin{aligned} &\frac{d}{dt}F(t,x,\dot{x})|_{(t_0,x_0,\dot{x}_0)} = 0, \\ &\quad\vdots \\ &\frac{d^k}{dt^k}F(t,x,\dot{x})|_{(t_0,x_0,\dot{x}_0)} = 0 \end{aligned} \tag{1.4b}$$

erfüllen.

Bemerkung 1.3.1 (Numerische Berechnung konsistenter Anfangswerte)
In der Praxis können bei der numerischen Berechnung konsistenter Anfangswerte zum einen Probleme dadurch entstehen, daß die Anfangswerte sowohl die Gleichungen des indexreduzierten Systems als auch der Invarianten erfüllen müssen. Dadurch sind in der Regel nicht mehr alle Anfangswerte der differentiellen Variablen frei. Zur Berechnung konsistenter Anfangswerte muß man die Anzahl der Freiheitsgrade des Systems und die freien Variablen kennen, um daraus die restlichen, abhängigen Variablen eindeutig bestimmen zu können. Das von Pantelides [Pan88] entwickelte Verfahren kann nur die Anzahl der strukturellen Freiheitsgrade und der dazugehörigen freien Variablen bestimmen, nicht aber die tatsächliche Anzahl der Freiheitsgrade und der tatsächlich freien Variablen.

Die in Abschnitt 1.1 erwähnten Verfahren umgehen dieses Problem, indem sie entweder (wie bei Bachmann et al. [BBMP90]) pro hinzugefügter Gleichung eine differentielle Gleichung, die zur Elimination der zeitlichen Ableitungen der Zustandsvariablen beigetragen hat, weglassen oder aber (wie bei Mattsson und Söderlind [MS93] und Pantelides et al. [PSV94]) sogenannte „dummy derivatives“ hinzufügen, um jeweils ein wohlbestimmtes System zu erhalten.

Weitere Probleme bei der numerischen Berechnung der konsistenten Anfangswerte können dadurch auftreten, daß die Gleichungen stark nichtlinear sind und für die algebraischen Variablen keine guten Startschätzungen vorliegen. Auf die numerische Behandlung dieser Problematik wird in Abschnitt 3.1 näher eingegangen.

1.4 Existenz und Eindeutigkeit

Für ein Anfangswertproblem einer gewöhnlichen Differentialgleichung der Form

$$\dot{y} = f(t, y), \quad y(t_0) = y_0 \tag{1.5}$$

lassen sich einfache Aussagen über die Existenz und Eindeutigkeit der Lösung treffen.

Satz 1.4.1 (Existenz und Eindeutigkeit bei ODE-Systemen)
Die Funktion $f : \mathbb{R} \times U \to \mathbb{R}^n$, $U \subseteq \mathbb{R}^n$ offen, aus (1.5) sei stetig auf U und genüge in U einer lokalen Lipschitz-Bedingung:

$\forall\, t \in \mathbb{R}$ und $y_0 \in U$ existiert eine Umgebung $\tilde{U} \equiv \tilde{U}(y_0)$ um y_0 und eine Konstante $L = L(y_0)$, so daß $\forall\, y \in \tilde{U}$ gilt:

$$\|f(t, y_0) - f(t, y)\| \leq L \|y_0 - y\| .$$

Dann besitzt das Anfangswertproblem (1.5) für jedes $t_0 \in \mathbb{R}$ und jedes $y_0 \in U$ eine Lösung und diese ist eindeutig bestimmt.

Beweis: siehe zum Beispiel das Buch von Walter [Wal80] über gewöhnliche Differentialgleichungen.

Im folgenden betrachten wir Anfangswertprobleme für linear implizite DAEs vom Index 1 der Form

$$\begin{aligned} A(t,y,z)\,\dot{y} &= f(t,y,z) && \text{(1.6a)} \\ 0 &= g(t,y,z)\,, \quad y(t_0)=y_0, z(t_0)=z_0\,. && \text{(1.6b)} \end{aligned}$$

Hierbei ist die Frage nach Existenz und Eindeutigkeit einer Lösung etwas komplizierter. Zum Beispiel können die Matrizen A oder g_z an manchen Punkten singulär sein (was zu sogenannten Impasse-Punkten oder Bifurkationen führen kann), oder die Anfangswerte sind inkonsistent.

Satz 1.4.2 (Existenz und Eindeutigkeit bei DAE-Systemen vom Index 1)
Seien $A : \mathbb{R} \times S \to \mathbb{R}^{n_y} \times \mathbb{R}^{n_y}$, $f : \mathbb{R} \times S \to \mathbb{R}^{n_y}$ und $g : \mathbb{R} \times S \to \mathbb{R}^{n_z}, n_y, n_z \geq 1, \mathcal{C}^r$-Funktionen, $r \geq 2$ und $S \subseteq \mathbb{R}^{n_y+n_z}$ offen.

Die Menge

$$S_0 = \left\{ (t,x) \in \mathbb{R} \times S : \; Rang \begin{pmatrix} A(t,x) & 0 & -f(t,x) \\ g_y(t,x) & g_z(t,x) & g_t(t,x) \end{pmatrix} = n_x := n_y + n_z \right\}$$

ist dann eine offene Teilmenge im $\mathbb{R}^{1+n_x}$.

Wir betrachten die Mannigfaltigkeit

$$\mathcal{M}(g,S) = \{ (t,x) \in \mathbb{R} \times S : \; g(t,x) = 0 \}\,.$$

Sei $\mathcal{M}_0 = \mathcal{M}(g,S) \cap S_0 \neq \emptyset$. Dann ist $\mathcal{M}_0$ eine Untermannigfaltigkeit von $\mathcal{M}(g,S)$ und zu jedem $(t_0,x_0) \in \mathcal{M}_0$ existiert eine $\mathcal{C}^{r-1}$-Lösung von (1.6) durch (t_0,x_0) und ist eindeutig.

Beweis: siehe die Untersuchungen von Rheinboldt aus dem Jahre 1984 [Rhe84] zu DAE-Systemen.

Bemerkung 1.4.1 (Numerische Lösung von DAEs vom Index 1)
Seien

$$\begin{aligned} S_1 &= \left\{ (t,x) \in \mathbb{R} \times S : \; Rang \begin{pmatrix} A(t,x) & 0 \\ g_y(t,x) & g_z(t,x) \end{pmatrix} = n \right\} \\ \text{und } \mathcal{M}_1 &= \mathcal{M}(g,S) \cap S_1 \neq \emptyset. \end{aligned} \qquad (1.7)$$

Für die numerische Lösung des DAE-Systems (1.6) mit Standard-Methoden muß für alle Punkte der Lösung zusätzlich $(t,x) \in \mathcal{M}_1$ gelten. Dann sind die Matrizen A und g_z für alle (t,x) auf der Mannigfaltigkeit $\mathcal{M}_1$ regulär, ihre Inversen beschränkt und die Anfangswerte $(t_0,x_0) \in \mathcal{M}_1$ konsistent, und das Anfangswertproblem (1.6) besitzt eine eindeutig bestimmte Lösung $x(t)$, die stetig (und $r-1$ mal differenzierbar) von den Anfangswerten y_0 abhängt (z_0 ist durch y_0 und $g(t_0,y_0,z_0)=0$ eindeutig bestimmt).

Bemerkung 1.4.2 (Impasse-Punkte)

Gilt Rang $\begin{pmatrix} A(t,x) & 0 \\ g_y(t,x) & g_z(t,x) \end{pmatrix} = n-1$ *für ein* $(t,x) \in \mathcal{M}_0$*, so exisitiert zwar die Lösung des DAE-Systems nach Satz 1.4.2 und ist eindeutig bestimmt, jedoch erhält man Probleme bei der numerischen Lösung. Solch ein Punkt wird Impasse-Punkt genannt. Für die numerische Lösung kann eine Augmentierung des Systems um die Variable t vorgenommen werden wie es zum Beispiel bei Rabier und Rheinboldt [RR94a, RR94b] oder Bauer [Bau94] beschrieben ist.*

Kapitel 2

Numerische Lösung von Anfangswertproblemen bei DAE-Systemen

Die Modellierung von chemischen Reaktionen und verfahrenstechnischen Prozessen führt in der Regel auf steife Systeme. Steifheit bedeutet, daß einige Komponenten der Lösungstrajektorie sehr stark abklingen, andere hingegen sich bezüglich der Zeit oder dem Ort t nur langsam ändern, also „steif“ sind.

Betrachtet man eine gewöhnliche Differentialgleichung der Form

$$\dot{y} = f(t, y), \tag{2.1}$$

so heißt dies, daß die Matrix $\partial f/\partial y$ einige Eigenwerte mit betragsmäßig sehr großen negativen Realteilen hat. Die Stabilitätsgebiete von expliziten Verfahren decken jedoch nur einen kleinen Bereich der negativen Halbachse ab. Dies stellt eine Beschränkung an die Schrittweite h dar mit $h \cdot |\mathrm{Re}\ \lambda_{max}| \leq const$, λ_{max} der Eigenwert von $\partial f/\partial y$ mit betragsmäßig größtem negativem Realteil. Falls $|\mathrm{Re}\ \lambda_{max}|$ sehr groß ist, müßte das Verfahren aus Stabilitätsgründen mit sehr kleinen Schrittweiten arbeiten.

Für die numerische Lösung von steifen Systemen werden deshalb implizite Verfahren verwendet. Dabei haben sich BDF-Verfahren wegen ihrer guten Stabilitätseigenschaften gerade zur Lösung der Systeme bewährt, bei denen zwar betragsmäßig große negative Realteile der Eigenwerte, nicht aber sehr große imaginäre Anteile auftreten. Es handelt sich dabei um Mehrschrittverfahren, die auf numerischer Differentiation durch sogenannte rückwärtige Differentiationsformeln (engl. backward differentiation formulae) beruhen.

BDF-Formeln wurden 1952 von Curtiss und Hirschfelder [CH52] eingeführt und vor allem durch die Untersuchungen von Gear 1971 [Gea71] und seine Erweiterung auf DAEs bekannt. Es folgten weitere Implementierungen wie zum Beispiel das von Petzold und Mitarbeitern entwickelte Programmpaket DASSL [Pet91, BCP96] und die daraus hervorgegangene Erweiterung DASPK [LP99a, LP99b] sowie der Code VODE von Brown

et al. [BBH89, BHB98]. Die Verfahren besitzen eine adaptive Ordnungs- und Schrittweitensteuerung, jedoch berücksichtigt die Schrittweitensteuerung in den erwähnten Codes nur zum Teil das tatsächliche nichtäquidistante Gitter. Außerdem werden in allen zur Zeit verwendeten Codes Strategien zur Reduzierung des Lineare-Algebra-Aufwands verwendet. Die Iterationsmatrix und ihre Zerlegung zur Lösung des nichtlinearen Gleichungssystems wird so lange wie möglich eingefroren.

Auch implizite Runge-Kutta-Verfahren haben sich zur Lösung von Anfangswertproblemen bei steifen ODE- und DAE-Systemen bewährt. Dabei werden in der Regel die sogenannten SDIRK-Verfahren (engl. singly diagonally implicit Runge-Kutta methods) verwendet, da sie zur Lösung der nichtlinearen Gleichungssysteme in jeder Stufe die gleiche Iterationsmatrix benötigen. Am bekanntesten sind das Verfahren SDIRK4 von Hairer und Wanner [HW96b] sowie die voll-impliziten Runge-Kutta-Verfahren RADAU [HW98] und RADAU5 [HW88, HW96a] von Hairer und Wanner (siehe auch [HW96b]). Auch diese besitzen Strategien zur adaptiven Schrittweitensteuerung und frieren die Iterationsmatrix nicht nur in jeder Stufe innerhalb eines Schrittes, sondern auch über mehrere Integrationsschritte hinweg ein.

Ein weiteres bekanntes Verfahren zur Lösung von Anfangswertproblemen in DAE-Systemen ist das Extrapolationsverfahren LIMEX von Deuflhard et al. [DHZ87] und seine Weiterentwicklung von Ehrig und Nowak [EN99]. Auch hier wird die Iterationsmatrix so lange wie möglich eingefroren, allerdings benötigt LIMEX insgesamt meist mehr Auswertungen und Zerlegungen der Iterationsmatrix als die anderen Integratoren.

Das im Rahmen dieser Arbeit weiterentwickelte Programmpaket DAESOL wurde in den letzten 10 bis 15 Jahren in der Arbeitsgruppe von Bock von Bleser [Ble86], Eich [Eic87] und Bauer [Bau94, BFD$^+$97, FBGS96, BBS99] entwickelt.

Wir betrachten zunächst die Konsistenz- und Konvergenzaussagen von linearen Mehrschrittverfahren und im speziellen von BDF-Verfahren auf äquidistantem Gitter und untersuchen, wie sich die Aussagen auf lineare Mehrschrittverfahren auf variablem Gitter und variabler Ordnung erweitern lassen. Anschließend leiten wir ein spezielles BDF-Verfahren her, wie es im Integrator DAESOL verwendet wird. Zuletzt gehen wir auf die speziellen Aspekte der Implementierung ein, wie etwa die Formeln zur Fehlerschätzung und Schrittweitensteuerung, die das tatsächliche nichtäquidistante Gitter berücksichtigen, und die Monitor-Strategie zur Reduzierung des Lineare-Algebra-Aufwands.

2.1 Theoretische Grundlagen bei BDF-Verfahren

Bei der numerischen Lösung von Differentialgleichungssystemen spielt zum einen der Diskretisierungsfehler, also der lokale Fehler in einem Schritt, aber auch der globale Fehler und somit die Konvergenz eines Verfahrens eine große Rolle.

Wir betrachten zunächst die allgemeine Klasse von linearen Mehrschrittverfahren zur numerischen Lösung von Anfangswertproblemen bei gewöhnlichen Differentialgleichungen

der Form

$$\dot{y} = f(t, y), \quad y(t_0) = y_0. \tag{2.2}$$

Hierfür sind Aussagen zur Konsistenzordnung, die das Verhalten des lokalen Fehlers beschreiben, aus der Literatur bekannt. Anschließend untersuchen wir die Stabilitätsgebiete von BDF-Verfahren und geben Formeln für ein BDF-Verfahren in Newton-Darstellung an.

2.1.1 Lineare Mehrschrittverfahren

Wir betrachten zunächst lineare Mehrschrittverfahren angewandt auf gewöhnliche Differentialgleichungen der Form (2.2) und untersuchen Konsistenz und Konvergenz dieser Verfahren.

Definition 2.1.1 (Lineare Mehrschrittverfahren)
Ein lineares Mehrschrittverfahren mit k Schritten *zur Lösung des Anfangswertproblems (2.2) für eine gewöhnliche Differentialgleichung im Schritt $n+1$ auf einem äquidistanten Gitter ist durch die Vorgabe der k Startwerte $y_n, \ldots, y_{n-k+1}$ und einer Differenzengleichung*

$$\sum_{i=0}^{k} \alpha_i y_{n+1-i} = h \cdot \sum_{i=0}^{k} \beta_i f(t_{n+1-i}, y_{n+1-i}), \tag{2.3}$$

mit $\alpha_i, \beta_i \in \mathbb{R}$, $i = 0, \ldots, k$, $\alpha_0 \neq 0$, $|\alpha_k| + |\beta_k| \neq 0$ und $t_j = t_0 + j\,h$, $j = 1, \ldots, n+1$, bestimmt.

Zur qualitativen Beurteilung des Verfahrens spielt der lokale Diskretisierungsfehler eine zentrale Rolle. Die Konsistenzordnung des Verfahrens gibt an, wie schnell der lokale Fehler für $h \to 0$ gegen Null strebt.

Definition 2.1.2 (Konsistenzordnung linearer Mehrschrittverfahren)
Ein lineares Mehrschrittverfahren der Form (2.3) hat die Konsistenzordnung *p, wenn für jede Funktion $y(t) \in \mathcal{C}^{p+1}[t_0, t_f]$ gilt*

$$L[y(t), h] := \sum_{i=0}^{k} (\alpha_i\, y(t + i\,h) - h\,\beta_i\, \dot{y}(t + i\,h)) = \mathcal{O}(h^{p+1}) \quad \text{für } h \to 0.$$

Zusätzlich zum lokalen Fehler des Verfahrens untersuchen wir die Fortpflanzung des Fehlers. Die Stabilität von linearen Mehrschrittverfahren wird über Eigenschaften des erzeugenden Polynoms

$$\rho(\xi) = \alpha_0 \xi^k + \alpha_1 \xi^{k-1} + \cdots + \alpha_{k-1}\xi + \alpha_k \tag{2.4}$$

beschrieben. Dieses gibt das Verhalten der Lösung für $n \to \infty$ bzw. für $h \to 0$ bei konstantem Faktor $n \cdot h$ an.

Definition 2.1.3 (Nullstabilität linearer Mehrschrittverfahren)
Ein lineares Mehrschrittverfahren der Form (2.3) heißt nullstabil, *wenn das erzeugende Polynom $\rho(\xi)$ der Wurzelbedingung genügt, das heißt*

1. *die Wurzeln von $\rho(\xi)$ liegen in oder auf dem Einheitskreis: $|\xi| \leq 1$ für $\rho(\xi) = 0$*
2. *und die auf dem Einheitskreis liegenden Wurzeln sind einfach.*

Wir beschränken uns im folgenden auf Stabilitätsuntersuchungen für die spezielle Klasse von BDF-Verfahren.

Definition 2.1.4 (BDF-Verfahren)
Ein BDF-Verfahren der Ordnung k *zur Lösung der gewöhnlichen Differentialgleichung (2.1) im Schritt $n+1$ auf einem äquidistanten Gitter ist durch die Vorgabe der k Startwerte $y_n, \ldots, y_{n-k+1}$ und einer Differenzengleichung*

$$\sum_{i=0}^{k} \alpha_i y_{n+1-i} = h \cdot f(t_{n+1}, y_{n+1}) \,, \tag{2.5}$$

mit $\alpha_i \in I\!R$, $i = 0, \ldots, k$, $\alpha_0 \neq 0$, $\alpha_k \neq 0$ und $t_j = t_0 + j\,h$, $j = 1, \ldots, n+1$ bestimmt.

Mit Hilfe von Definition 2.1.3 läßt sich die Stabilität von BDF-Verfahren zeigen.

Satz 2.1.1 (Nullstabilität von BDF-Verfahren)
Das BDF-Verfahren (2.5) vom Grade k ist nullstabil für $k \leq 6$ und instabil für $k \geq 7$.

Beweis: Man zeigt, daß das erzeugende Polynom $\rho(\xi)$ die einfache Nullstelle $\xi = 1$ hat und alle anderen Nullstellen innerhalb des Einheitskreises liegen. Der Beweis wurde zum ersten Mal von Cryer 1972 [Cry72] veröffentlicht. Beweise finden sich zum Beispiel auch bei Grigorieff [Gri77] und Hairer et al. [HNW93].

Von Dahlquist wurde 1956 [Dah56] eine Beziehung zwischen der Konsistenz und Stabilität eines Verfahrens und dessen Konvergenz hergestellt.

Satz 2.1.2 (Konvergenz von linearen Mehrschrittverfahren)
Das lineare Mehrschrittverfahren (2.3) ist konvergent von der Ordnung p genau dann, wenn es konsistent ist von der Ordnung p und nullstabil.

Hiermit ist auch die Konvergenz von BDF-Verfahren bis zur Ordnung 6 gezeigt.

Satz 2.1.3 (Konvergenz von BDF-Verfahren)
Das BDF-Verfahren (2.5) vom Grade k, $k \leq 6$, ist konvergent von der Ordnung $p = k$, wenn die zurückliegenden k Startwerte von der Genauigkeit $\mathcal{O}(h^k)$ sind.

2.1.2 Lineare Mehrschrittverfahren auf variablem Gitter

Die Konsistenz- und Konvergenzaussagen aus Abschnitt 2.1.1 gelten zunächst nur für Mehrschrittverfahren auf äquidistantem Gitter. Jedoch arbeiten praxistaugliche Verfahren nicht auf äquidistantem Gitter, sondern passen die Schrittweite an die Nichtlinearität des Systems, die aktuelle Ordnung des Verfahrens und an die vorgegebene Genauigkeit an. Im folgenden untersuchen wir, wie sich die Konsistenz- und Konvergenzaussagen auf Verfahren auf variablem Gitter übertragen lassen.

Definition 2.1.5 (Lineare Mehrschrittverfahren auf variablem Gitter)
Ein lineares Mehrschrittverfahren mit k Schritten auf einem variablen Gitter $I_h = \{t_0, \ldots, t_{n+1}\}$, $t_i < t_{i+1}, i = 0, \ldots, n$, $h_{i+1} = t_{i+1} - t_i$, hat die Gestalt

$$\sum_{i=0}^{k} \alpha_{i,n+1}\, y_{n+1-i} = h_{n+1} \sum_{i=0}^{k} \beta_{i,n+1}\, f(t_{n+1-i}, y_{n+1-i}),\ n = 0, \ldots, N, \tag{2.6}$$

wobei die k Startwerte $y_n, \ldots, y_{n-k+1}$ gegeben sind sowie die Koeffizienten $\alpha_{i,n+1}$ und $\beta_{i,n+1}$ mit

$$\alpha_{i,n+1}, \beta_{i,n+1} \in I\!R,\ |\alpha_{k,n+1}| + |\beta_{k,n+1}| \neq 0,$$

die von den Quotienten der Schrittweiten $\omega_j = h_j/h_{j-1}, j = n+1-k, \ldots, n+1$, abhängen.

Analog zu den Mehrschrittverfahren auf äquidistantem Gitter definieren wir die Konsistenzordnung des Verfahrens auf variablem Gitter.

Definition 2.1.6 (Konsistenzordnung auf variablem Gitter)
Ein lineares Mehrschrittverfahren mit k Schritten auf einem variablen Gitter hat die Konsistenzordnung *p, wenn für alle Polynome $P(t)$ vom Grade kleiner gleich p und für alle Gitter I_h gilt*

$$\sum_{i=0}^{k} \alpha_{i,n+1}\, P(t_{n+1-i}) = h_{n+1} \sum_{i=0}^{k} \beta_{i,n+1}\, P'(t_{n+1-i})\,.$$

Für den lokalen Diskretisierungsfehler gilt der

Satz 2.1.4 (Lokaler Diskretisierungsfehler auf variablem Gitter)
Das lineare Mehrschrittverfahren habe die Konsistenzordnung p. Ferner gelte

1. *Die Quotienten der Schrittweiten $\omega_n = h_n/h_{n-1}$ seien für alle n beschränkt.*
2. *Die Koeffizienten $\alpha_{i,n+1}$ und $\beta_{i,n+1}$, $i = 1, \ldots, k$ seien in jedem Schritt $n+1, n = 0, \ldots, N-1$, beschränkt.*

Dann genügt unter der Voraussetzung $f(t, y) \in \mathcal{C}^p(I\!R \times S)$ der lokale Diskretisierungsfehler der asymptotischen Beziehung $\mathcal{O}(h^{p+1})$, $h = max_i\, h_i$.

Beweis: Taylorreihenentwicklung von $y(t_{n+1})$ führt auf

$$\sum_{i=0}^{k} \alpha_i \, y(t_{n+1-i}) - h_{n+1} \sum_{i=0}^{k} \beta_i \, \dot{y}(t_{n+1-i}) = \mathcal{O}(h_{n+1}^{p+1}).$$

Bei den Untersuchungen zur Stabilität von linearen Mehrschrittverfahren auf variablem Gitter wird das Verfahren als Störung des linearen Mehrschrittverfahrens auf äquidistantem Gitter aufgefaßt. Im folgenden betrachten wir eine Stabilitätsaussage für lineare Mehrschrittverfahren auf variablem Gitter von Crouzeix und Lisbona [CL84].

Satz 2.1.5 (Stabilität auf variablem Gitter)
Das lineare Mehrschrittverfahren (2.6) erfülle die folgenden Bedingungen:

1. *das Verfahren habe die Konsistenzordnung p,*
2. *die Koeffizienten $\alpha_{i,n+1} \equiv \alpha_{i,n+1}(\omega_{n+1}, \ldots, \omega_{n+1-k})$ sind stetig in einer Umgebung von (1,. . . , 1) und*
3. *das Verfahren sei auf äquidistantem Gitter stark stabil, das heißt alle Wurzeln von*

$$\rho(\xi) = \sum_{i=0}^{k} \alpha_{i,n+1}(1, \ldots, 1)\xi^i = 0$$

liegen im Inneren des Einheitskreises mit der Ausnahme von $\xi_1 = 1$.

Dann existieren reelle Zahlen ω, Ω, $\omega < 1 < \Omega$, so daß das Verfahren stabil ist, falls

$$\omega \leq h_{n+1}/h_n \leq \Omega \quad \text{für alle } n.$$

Beweis: Das Mehrschrittverfahren auf variablem Gitter wird als Störung des Verfahrens auf äquidistantem Gitter betrachtet. Für hinreichend kleine Störungen können die Stabilitätsbedingungen auf das Verfahren auf variablem Gitter übertragen werden.

Als letzten Schritt untersuchen wir das Konvergenzverhalten der Mehrschrittverfahren auf variablem Gitter.

Satz 2.1.6 (Konvergenz auf variablem Gitter)
Das lineare Mehrschrittverfahren (2.6) erfülle die folgenden Bedingungen:

1. *das lineare Mehrschrittverfahren (2.6) sei stabil, von der Ordnung p und die Koeffizienten $\alpha_{i,n+1}$ und $\beta_{i,n+1}$ seien bechränkt;*
2. *die Anfangswerte erfüllen die Bedingung $\|y(t_i) - y_i\| = \mathcal{O}(h^p)$, $i = 0, \ldots, p-1$;*

3. die Quotienten der Schrittweiten seien für alle n beschränkt: $h_n/h_{n-1} \leq \Omega$.

Dann konvergiert das Verfahren von der Ordnung p, das heißt für jede Differentialgleichung $\dot{y}(t) = f(t,y)$, $y(t_0) = y_0$, und f hinreichend oft differenzierbar, gilt für den globalen Diskretisierungsfehler

$$\|y(t_{n+1}) - y_{n+1}\| \leq C\, h^p,\ t_{n+1} \in [t_0, t_f],\ h = \max_i h_i\,.$$

Beweis: Mit

$$Y_{n+1} = \begin{pmatrix} y_{n+1} \\ \vdots \\ y_{n-k+1} \end{pmatrix},\ A_{n+1} = \begin{pmatrix} -\tilde{\alpha}_{1,n+1} & & \cdots & & -\tilde{\alpha}_{k,n+1} \\ 1 & 0 & & \cdots\ 0 & 0 \\ 0 & 1 & 0 & \cdots\ 0 & 0 \\ \vdots & \ddots & \ddots & \ddots\ \vdots & \vdots \\ 0 & \cdots & 0 & 1\ 0 & 0 \\ 0 & & \cdots & 0\ 1 & 0 \end{pmatrix},\ \tilde{\alpha}_{i,n+1} = \frac{\alpha_{i,n+1}}{\alpha_{0,n+1}}$$

und

$$\psi_{n+1} = \frac{1}{\alpha_{0,n+1}}\, f\left(t_{n+1}, h_{n+1}\,\psi_{n+1} - \sum_{i=1}^{k} \tilde{\alpha}_{i,n+1}\, y_{n+1-i}\right)$$

schreiben wir das Verfahren (2.6) verkürzt als

$$Y_{n+1} \;=\; (A_{n+1} \otimes I)\ Y_n + h_{n+1}\,\Phi_{n+1},$$

mit $\Phi_{n+1} = (e_1 \otimes I)\ \psi_{n+1}$, $e_1^T = (1, 0, \ldots, 0)$. Somit gilt für den globalen Fehler

$$\begin{aligned} Y(t_{n+1}) - Y_{n+1} \;=\; & ((A_{n+1} \cdots A_1) \otimes I)\ (Y(t_0) - Y_0) \\ & + \sum_{j=1}^{n+1} h_{j+1}\ ((A_{n+1} \cdots A_{j+1}) \otimes I)\ (\Phi(t_j) - \Phi_j) \\ & + \sum_{j=1}^{n+1} ((A_{n+1} \cdots A_{j+1}) \otimes I)\ \delta_{j+1}\,, \end{aligned}$$

wobei δ_{j+1} den lokalen Fehler im Schritt $j+1$ beschreibt

$$\delta_{j+1} = Y(t_{j+1}) - (A_{n+1} \otimes I) Y(t_j) - h_{j+1}\,\Phi(t_j)\,.$$

Dabei bezeichne $Y(t_j)$ die analytische Lösung von Y zum Zeitpunkt t_j und $\Phi(t_j) := (e_1 \otimes I)\psi(t_j)$, $\psi(t_j) = \frac{1}{\alpha_{0,j}}\, f\left(t_j, h_j\,\psi(t_j) - \sum_{i=1}^{k} \tilde{\alpha}_{i,j}\, y(t_{j-i})\right)$. Aufgrund der Stabilität des Mehrschrittverfahrens sind die Produkte von A beschränkt und der lokale Diskretisierungsfehler genügt der Beziehung $\delta_{j+1} = \mathcal{O}(h_{j+1}^{p+1})$. Da f als Lipschitz-stetig vorausgesetzt ist, ist auch Φ Lipschitz-stetig mit Lipschitz-Konstante L und man zeigt die Konvergenz per Induktion:

$$\|Y(t_0) - Y_0\| = \epsilon_0,$$

$$
\begin{aligned}
&\|Y(t_n) - Y_n\| = \epsilon_n \\
\Longrightarrow \|Y(t_{n+1}) - Y_{n+1}\| =: \epsilon_{n+1} &\leq h_n\, L\, \epsilon_n + C\, h^p \\
&\leq \sum_{i=0}^{n} h_i\, L\, \epsilon_i + C_1\, h^p,\ h = \max_i h_i
\end{aligned}
$$

Die Stabilität von BDF-Verfahren auf variablem Gitter gilt nach Satz 2.1.5 nur für eine Folge von Schrittweiten, deren Quotienten innerhalb der Schranken $\omega \leq h_i/h_{i-1} \leq \Omega$ liegen und nicht allzu weit von einer äquidistanten Folge abweichen. Grigorieff [Gri83] hat Schranken ω und Ω berechnet, für die die Stabilität von BDF-Verfahren höherer Ordnung garantiert wird.

k	2	3	4	5
ω	0	0.836	0.979	0.997
Ω	2.414	1.127	1.019	1.003

Tabelle 2.1: Schranken ω und Ω an die Quotienten der Schrittweiten für BDF-Verfahren auf variablem Gitter

Bemerkung 2.1.1 (Beschränkungen an die Schrittweitenfolge)
Die Schranken aus Tabelle 2.1 lassen für Ordnungen ≥ 4 kaum Spielraum für Schrittweitenänderungen. Sie sind allerdings auch sehr pessimistisch, da sie alle möglichen Varianten der Schrittweitenfolge berücksichtigen. So ist zum Beispiel das BDF-Verfahren der Ordnung 4 bei einer gleichmäßigen Vergrößerung der Schrittweite um den Faktor 1.5 immer noch stabil (das erzeugende Polynom $\rho(\xi)$ aus (2.4) hat die einfache Wurzel $\xi = 1$ und alle anderen Wurzeln liegen für $k = 4$ und bei Vergrößerung der Schrittweite um den konstanten Faktor 1.5 innerhalb des Einheitskreises). Die Schranken stellen somit nur eine notwendige, nicht aber eine hinreichende Bedingung dar und werden in der Praxis meist nicht eingehalten. Die in DAESOL implementierte Schrittweitensteuerung strebt eine gleichmäßige Änderung der Schrittweiten an und versucht, große Änderungen zu vermeiden. Die Schranken aus obiger Tabelle müssen dabei nicht erfüllt werden.

Etwas weniger pessimistische Ergebnisse wurden von Gear und Tu [GT74] gezeigt, die eine stetige Änderung der Schrittweiten voraussetzen.

Calvo et al. [CLM87] untersuchten die Stabilität für BDF-Verfahren auf variablem Gitter für sogenannte pseudo-äquidistante Verfahren wie sie von Nordsieck [Nor62] vorgeschlagen wurden. Die Methode auf äquidistantem Gitter bleibt dabei an sich erhalten. Bei Schrittweitenänderungen werden zurückliegende Werte für das neue Gitter über Polynominterpolation ermittelt. Sie berechneten Schranken, die bei einer Änderung der Schrittweiten innerhalb dieser Schranken Stabilität garantieren. Dabei werden die Intervalle für die Schrittweitenänderungen größer, je mehr Schritte im Anschluß an eine Änderung konstant gehalten werden. Eine Schrittweitensteuerung, die die Schrittweite nach einer Änderung

über mehrere Schritte konstant hält, findet sich auch heute noch in vielen gängigen Codes, auch wenn keine pseudo-äquidistanten Verfahren verwendet werden.

2.1.3 Lineare Mehrschrittverfahren variabler Ordnung und Schrittweite

Bisher wurden nur Änderungen der Schrittweite bei linearen Mehrschrittverfahren betrachtet, Voraussetzung war jedoch, daß die k zurückliegenden Werte von der Ordnung k sind.

Da man bei der numerischen Integration von Anfangswertproblemen der Form (2.2) in der Regel nur den Anfangswert, nämlich $y(t_0) = y_0$, gegeben hat, verwenden die meisten Integratoren eine der beiden folgenden Möglichkeiten für die Startphase:

1. Berechne die k Startwerte mit einem Einschrittverfahren höherer Ordnung, zum Beispiel einem Runge-Kutta-Verfahren. Wenn wir für die folgende Integration für das Mehrschrittverfahren die gleiche Ordnung verwenden, so müssen nur die Voraussetzungen aus Abschnitt 2.1.2 an die Schrittweitenänderungen erfüllt sein und die Konvergenz des Verfahrens von der Ordnung des Einschrittverfahrens ist garantiert. Allerdings darf das Mehrschrittverfahren für die gegebenen Voraussetzungen die vom Einschrittverfahren vorgegebene Ordnung nicht mehr verändern.

2. Man startet mit niedriger Ordnung m (in der Regel 1, das heißt implizites Euler-Verfahren bei BDF-Verfahren) und kleiner Schrittweite und erhöht langsam sowohl die Ordnung als auch die Schrittweite. Sind die Startwerte von der Ordnung m und das Verfahren von der Ordnung k, so ist die Lösung nach den theoretischen Untersuchungen aus Abschnitt 2.1.2 konvergent von der Ordnung $\min(m, k)$. Das heißt, daß der Fehler weiterhin der Beziehung $\mathcal{O}(h^{m+1})$ genügt, auch wenn die Ordnung anschließend erhöht wurde.

Da auch im ersten Fall die Integration mit konstanter Ordnung eine bedeutende Einschränkung ist (zum einen, weil das Einschrittverfahren eventuell nicht die maximale Ordnung bietet, zum anderen weil manchmal auch eine Verkleinerung der Ordnung etwa bei niedrigen Genauigkeiten sinnvoll sein kann), muß der Einfluß von Ordnungsänderungen auf die Stabilität der Mehrschrittverfahren untersucht werden.

Gear und Watanabe [GW74] untersuchten Stabilität und Konvergenz bei Änderung der Ordnung von Mehrschrittverfahren, sogar allgemeiner bei Änderung der Methode mit verschiedenen Ordnungen, zum Beispiel auch zum Hin- und Herschalten zwischen steifen und nicht-steifen Lösern. Sie haben gezeigt, daß aus Stabilitätsgründen die Methode nach einer Änderung für ein paar Schritte konstant gehalten werden muß. Für die BDF-Verfahren bedeutet dies, daß die Ordnung nach Ordnungsänderungen für einige Schritte konstant gehalten werden muß. Allerdings untersuchten sie nicht die Konvergenzordnung des Fehlers bei Änderungen der Methoden.

Shampine und Zhang [SZ90] geben eine Schrittweiten- und Ordnungssteuerung für die Startphase an, wie sie für viele Integratoren verwendet wird. In jedem Schritt wird die aktuelle Schrittweite h so gewählt, daß der lokale Fehler kleiner oder gleich einer vorgegebenen Toleranz TOL ist. Der lokale Fehler ist bei unterschiedlichen Ordnungen m und k proportional zu h^{m+1} beziehungsweise H^{k+1}, jedoch werden die Schrittweiten h und H jeweils so gewählt, daß der lokale Fehler kleiner oder gleich TOL ist. Somit ist der globale Fehler – Stabilität vorausgesetzt – immer proportional zu TOL. In der Praxis hat sich jedoch gezeigt, daß der lokale Fehler bei niedrigeren Ordnungen nicht so gut approximiert wird. Fehler aus der Startphase der Integration mit niedriger Ordnung können noch für eine Weile den gesamten globalen Fehler dominieren, auch wenn die Ordnung bereits erhöht wurde. Dies kann vor allem dann passieren, wenn das System auch weniger steife Komponenten enthält. In Abschnitt 3.2 gehen wir näher auf die Fortpflanzung des in der Startphase mit niedrigeren Ordnungen gemachten Fehlers ein und leiten ein implizites Runge-Kutta-Verfahren her, das Werte höherer Ordnung generiert, die beim anschließenden Start des BDF-Verfahrens als zurückliegende Werte dienen.

2.1.4 BDF-Formeln in Newton-Darstellung

Im folgenden leiten wir ein BDF-Verfahren her, das die Konsistenz- und Stabilitätsbedingungen aus den vorhergehenden Abschnitten erfüllt. Wir beschränken uns zunächst wieder auf Anfangswertprobleme für gewöhnliche Differentialgleichungen der Form (2.2). Die Interpolationspolynome werden dabei in Newton'scher Form dargestellt. Dies erlaubt ein einfaches Abspeichern und Aufdatieren der benötigten Größen von einem Schritt zum nächsten.

Wir konstruieren zunächst ein Polynom $P^C_{n+1}(t)$, das die Konsistenzbedingungen erfüllt. Das Polynom interpoliert die zurückliegenden, bereits berechneten Werte von y

$$P^C_{n+1}(t_{n+1-i}) = y_{n+1-i}, \quad i = 1, \ldots, k,$$

und an der Stelle t_{n+1} ist die zeitliche Ableitung gleich der rechten Seite der ODE an der Stelle t_{n+1}

$$\dot{P}^C_{n+1}(t_{n+1}) = f(t_{n+1}, P^C_{n+1}(t_{n+1}))\,.$$

Diese $k+1$ Bedingungen bestimmen ein eindeutiges Polynom P^C_{n+1} vom Grade k. Der unbekannte Wert y_{n+1} an der Stelle t_{n+1} kann dargestellt werden durch

$$P^C_{n+1}(t_{n+1}) = y_{n+1}.$$

Dabei sei $y(t_{n+1})$ die wahre Lösung des Anfangswertproblems (2.2) im Schritt $n+1$ zum Zeitpunkt t_{n+1} und y_{n+1} die mit Hilfe des Diskretisierungsverfahrens approximierte Lösung.

In jedem Schritt wird also $\dot{y}$ durch die zeitliche Ableitung des Polynoms P^C_{n+1} approximiert

$$\begin{aligned}\dot{P}^C_{n+1}(t_{n+1}) =: & -\frac{1}{h}\left(\alpha_0 P^C_{n+1}(t_{n+1}) + \sum_{i=1}^{k} \alpha_i P^C_{n+1}(t_{n+1-i})\right) \\ := & -\frac{1}{h}\left(\alpha_0 y_{n+1} + \sum_{i=1}^{k} \alpha_i y_{n+1-i}\right) = f(t_{n+1}, y_{n+1})\,.\end{aligned} \tag{2.7}$$

Wir erhalten ein nichtlineares Gleichungssystem

$$F(y_{n+1}) = \alpha_0 y_{n+1} + h \cdot f(t_{n+1}, y_{n+1}) + \sum_{i=1}^{k} \alpha_i y_{n+1-i} \stackrel{!}{=} 0$$

in den Unbekannten y_{n+1}. Zur Lösung verwendet man zum Beispiel Newton-ähnliche Verfahren, die sich sehr gut zur Lösung bei steifen Systemen bewährt haben. Das speziell in DAESOL verwendete vereinfachte Newton-Verfahren mit Monitor-Strategie, das insbesondere die Anzahl der Auswertungen von A_y, f_y und g_y reduziert, wird in Abschnitt 2.3.1 näher erläutert.

Zur Berechnung der Lösung mit Newton-Verfahren benötigt man zusätzlich einen Startwert. Dieser wird dadurch gewonnen, daß man die letzten $k+1$ Werte von y interpoliert

$$P^P_{n+1}(t_{n+1-i}) \;=\; y_{n+1-i}, \quad i = 1, \ldots, k+1,$$

und dieses Polynom im Punkt t_{n+1} extrapoliert

$$P^P_{n+1}(t_{n+1}) \;=\; y^P_{n+1}.$$

Das Prädiktorpolynom, das die zurückliegenden Werte $y_n, \ldots, y_{n+1-k}$ interpoliert, ist gegeben durch

$$P^P_{n+1}(t_{n+1}) = \sum_{i=0}^{k} p_i(t_{n+1}) \nabla^i y_n$$

mit

$$p_i(t) = \begin{cases} 1 & i = 0 \\ \prod_{j=1}^{i}(t - t_{n+1-j}) & i = 1, \ldots, k+1 \end{cases}$$

und den dividierten Differenzen

$$\begin{aligned}\nabla^0 y_n =& y_n \\ \nabla^i y_n =& \frac{\nabla^{i-1} y_n - \nabla^{i-1} y_{n-1}}{t_n - t_{n-i}}\,.\end{aligned} \tag{2.8}$$

An Stelle der dividierten Differenzen selbst werden sogenannte modifizierte dividierte Differenzen abgespeichert, was den Aufwand für Speicherplatz und für Updates von einem

Schritt zum nächsten reduziert. Um die Prädiktor- und Korrektorpolynome in Abhängigkeit der modifizierten dividierten Differenzen darstellen zu können, benötigen wir noch folgende Größen:

$$\begin{aligned} \psi_j(n+1) &= t_{n+1} - t_{n+1-j} = \psi_{j-1}(n) + h_{n+1} \\ \gamma_j(n+1) &= \sum_{i=1}^{j-1} \frac{1}{\psi_i(n+1)} \,. \end{aligned}$$

Die modifizierten dividierten Differenzen werden definiert als

$$\Phi_j^*(n) = \psi_1(n+1) \cdot \ldots \cdot \psi_{j-1}(n+1) \nabla^{j-1} y_n$$

und

$$\delta_j(n+1) = \sum_{i=1}^{j} \Phi_i^*(n)$$

beschreibt ihre Summe.

Der Prädiktor ist dann von der Form

$$\begin{aligned} y_{n+1}^P &= P_{n+1}^P(t_{n+1}) \\ &= \sum_{j=0}^{k} p_j(t_{n+1}) \nabla^j y_n \\ &= \sum_{j=0}^{k} \psi_1(n+1) \cdot \ldots \cdot \psi_j(n+1) \nabla^j y_n \\ &= \sum_{j=0}^{k} \Phi_{j+1}^*(n) \\ &= y_n + \sum_{j=1}^{k} \Phi_{j+1}^*(n) \,. \end{aligned} \tag{2.9}$$

Mit der Ableitung von p_j im Punkt t_{n+1}

$$\begin{aligned} \dot{p}_j(t_{n+1}) &= \frac{d}{dt} \left(\prod_{i=1}^{j} t_{n+1} - t_{n+1-i} \right) \\ &= \sum_{l=1}^{j} \left(\prod_{\substack{i=1 \\ i \neq l}}^{j} t_{n+1} - t_{n+1-i} \right) \\ &= \sum_{l=1}^{j} \left(\frac{1}{t_{n+1} - t_{n+1-l}} \right) \cdot \prod_{i=1}^{j} t_{n+1} - t_{n+1-i} \end{aligned}$$

$$
\begin{aligned}
&= p_j(t_{n+1}) \sum_{l=1}^{j} \frac{1}{\psi_l(n+1)} \\
&= p_j(t_{n+1}) \gamma_j(n+1)
\end{aligned}
$$

erhalten wir für den Korrektor

$$
\begin{aligned}
\dot{y}^C_{n+1} &= \dot{P}^C_{n+1}(t_{n+1}) \\
&= \gamma_{k+1}(n+1)\, y^C_{n+1} - \sum_{j=1}^{k} \frac{1}{\psi_j(n+1)} \delta_j(n+1) \\
&=: \gamma_{k+1}(n+1)\, y^C_{n+1} + c, \quad c = -\sum_{j=1}^{k} \frac{1}{\psi_j(n+1)} \delta_j(n+1)\,.
\end{aligned}
\tag{2.10}
$$

Bemerkung 2.1.2 (Auswertung der Interpolationspolynome)
Betrachtet man das Prädiktor- und Korrektorpolynom, so sieht man, daß die Terme zur Berechnung des Prädiktorpolynoms in die Berechnung des Korrektorpolynoms eingehen. Die Berechnung des Prädiktors erfordert somit keinen zusätzlichen Aufwand für das Verfahren.

2.2 BDF-Verfahren angewandt auf DAEs

Das Programmpaket DAESOL löst Anfangswertprobleme für linear implizite DAEs der Form

$$
\begin{aligned}
A(t,y,z)\,\dot{y} &= f(t,y,z) && \text{(2.11a)} \\
0 &= g(t,y,z), && \text{(2.11b)} \\
y(t_0) &= y_0, && \text{(2.11c)}
\end{aligned}
$$

die den Bedingungen (1.7) genügen, das heißt insbesondere, daß $A(t,y,z)$ und $\frac{\partial g}{\partial z}(t,y,z)$ entlang der Lösungstrajektorie regulär sein müssen. Die DAE heißt dann vom differentiellen Index 1 (siehe Definition 1.1.1).

Approximiert man die Ableitung $\dot{y}$ durch die Ableitung des Korrektorpolynoms so erhält man im Schritt $n+1$

$$
\begin{aligned}
A(t_{n+1}, y_{n+1}, z_{n+1})\,(\alpha_0 y_{n+1} + h\,c) &= -\,h\, f(t_{n+1}, y_{n+1}, z_{n+1}) \\
0 &= g(t_{n+1}, y_{n+1}, z_{n+1})\,, \\
\alpha_0 &= -\,h \cdot \gamma_{k+1}(n+1)\,.
\end{aligned}
\tag{2.12}
$$

Der Term c beschreibt dabei den konstanten Anteil der Ableitung des Korrektorpolynoms (2.10). Die Unbekannten $(y_{n+1}, z_{n+1}) =: x_{n+1}$ sind implizit durch das nichtlineare Gleichungssystem (2.12) definiert. Zur Lösung verwenden wir ein vereinfachtes Newton-Verfahren. Eine erste Schätzung für die Unbekannten x_{n+1} erhalten wir durch Auswertung

des Prädiktorpolynoms zum Zeitpunkt t_{n+1} für alle Komponenten von x. Diese dient als Startwert für das vereinfachte Newton-Verfahren.

Schreibt man für (2.12) kurz

$$F(x_{n+1}) = 0, \qquad \text{mit} \quad x_{n+1} = (y_{n+1}, z_{n+1}),$$

so definieren wir einen Newton-Schritt durch

$$x_{n+1}^{(m+1)} = x_{n+1}^{(m)} + \Delta x_{n+1}^{(m)}.$$

$\Delta x_{n+1}^{(m+1)}$ löst das lineare Gleichungssystem

$$J(x_{n+1}^{(m)}) \cdot \Delta x_{n+1}^{(m)} = -\, F(x_{n+1}^{(m)}), \tag{2.13}$$

wobei

$$J(x_{n+1}^{(m)}) = \begin{pmatrix} \alpha_0 A + A_y\,(\alpha_0 y_{n+1}^{(m)} + h\,c) + h\,f_y & A_z\,(\alpha_0 y_{n+1}^{(m)} + h\,c) + h\,f_z \\ g_y & g_z \end{pmatrix} \tag{2.14}$$

die Jacobi-Matrix von F beschreibt und c den konstanten Anteil der Ableitung des Korrektorpolynoms, die $\dot{y}_{n+1}$ approximiert.

Bemerkung 2.2.1 (Berechnung der Ableitungen von A)
Bei der Berechnung der Jacobi-Matrix mit numerischen Differenzen wird der Ausdruck $A_x(t_{n+1}, y_{n+1}^{(m)}, z_{n+1}^{(m)}) \cdot (\alpha_0 y_{n+1}^{(m)} + h\,c)$ direkt über Richtungsableitungen ausgewertet. Dies spart Rechenzeit und Speicherplatz. Die direkte Auswertung der Richtungsableitungen wird sowohl bei der Berechnung mit Hilfe numerischer Differenzen als auch bei der Berechnung mit Automatischer Differentiation berücksichtigt.

2.3 Algorithmen und Strategien in DAESOL

2.3.1 Lösung des nichtlinearen Gleichungssystems – Monitor-Strategie

In der Regel ist die Auswertung und Zerlegung der Jacobi-Matrix J aus (2.14) sehr aufwendig und benötigt den größten Teil der Rechenzeit während der Integration. Das gilt vor allem für große DAE-Systeme oder wenn die Auswertungen der Ableitungen der Rechte-Seite-Funktionen beziehungsweise der Matrix A sehr komplex sind.

In vielen Fällen ändert sich die Jacobi-Matrix von einer Newton-Iteration zur nächsten und sogar über mehrere BDF-Schritte hinweg nur wenig. Um Rechenzeiten einzusparen soll die Jacobi-Matrix deshalb so lange wie möglich eingefroren werden. Dies führt zwar zu etwas schlechteren Konvergenzeigenschaften des vereinfachten Newton-Verfahrens, der

Mehraufwand von ein oder zwei Newton-Iterationen steht aber meist in keinem Vergleich zu einer Auswertung und Zerlegung der Jacobi-Matrix.

Im folgenden leiten wir eine Monitor-Strategie her, die die Jacobi-Matrix oder Teile davon so lange wie möglich einfriert, gleichzeitig aber darauf achtet, daß das vereinfachte Newton-Verfahren innerhalb weniger Schritte konvergiert. Dabei wird insbesondere darauf geachtet, daß die Auswertung der Ableitungen der Modellfunktionen meist bedeutend aufwendiger ist als eine Zerlegung der Iterationsmatrix. Diese Strategie wurde von Bock und Eich eingeführt und zum ersten Mal in [Eic87] dokumentiert.

Wir untersuchen zunächst das Konvergenzverhalten des vereinfachten Newton-Verfahrens, worüber der folgende Satz (siehe Bock [Boc87]) Aussagen gibt.

Satz 2.3.1 (Lokaler Kontraktionssatz)
Sei $D \subseteq \mathbb{R}^n$ offen und $F : D \to \mathbb{R}^{n_x}$ eine $\mathcal{C}^1$-Funktion. Wir bezeichnen mit $J = \partial F/\partial x$ die Jacobi-Matrix von F und mit $\tilde{J}^{-1}$ eine Approximation der Inversen von J.

Für alle $\tau \in [0,1]$, $x, x+\Delta x \in D$ mit $\Delta x = -\tilde{J}^{-1}\, F(x)$ und für alle m existieren Schranken ω und κ, so daß:

i) $\|\tilde{J}^{-1}(J(x^m + \tau\Delta x^m) - J(x^m))\, \Delta x^m\| \le \omega^m\, \tau\, \|\Delta x^m\|^2, \quad \omega^m \le \omega < \infty$

ii) $\|\tilde{J}^{-1}(F(x^m) - J(x^m)\Delta x^m)\| \le \kappa^m \|\, \Delta x^m\|, \quad \kappa^m \le \kappa < 1,$

iii) der Startwert x^0 der Iteration erfülle die folgende Bedingung

$$\delta_0 := \frac{\omega^0}{2}\|\Delta x^0\| + \kappa^0 < 1 \tag{2.15}$$

iv) und die Kugel $D_0 := S_r(x^0)$ um x^0 mit Radius $r = \frac{\|\Delta x^0\|}{1-\delta_0}$ liege in D.

Dann gilt:

1. *Die Iteration $x^{m+1} = x^m + \Delta x^m$, $\Delta x^m = -\tilde{J}^{-1}\, F(x^m)$ ist wohldefiniert und bleibt in D_0.*
2. *Es existiert eine Nullstelle $x^* \in D_0$, gegen die x^m konvergiert.*
3. *Die Folge x^m konvergiert mindestens linear mit*

$$\|\Delta x^m\| \le \left(\frac{\omega^{m-1}}{2}\|\Delta x^{m-1}\| + \kappa^{m-1}\right) \|\Delta x^{m-1}\| \le \|\Delta x^{m-1}\|$$

4. *und für die m-te Iterierte gilt die a priori-Abschätzung*

$$\|x^m - x^*\| \le \|\Delta x^0\| \frac{\delta_0^m}{1-\delta_0}\,. \tag{2.16}$$

Beweis: Der Beweis erfolgt in Anlehnung an den Beweis des Banach'schen Fixpunktsatzes, siehe Bock [Boc87].

Der Faktor ω beschreibt dabei die Nichtlinearität der Jacobi-Matrix J und κ die Güte der Näherungsinversen $\tilde{J}^{-1}$.

Wir brechen das Newton-Verfahren ab, wenn das Inkrement $\|\Delta x^m\|$ kleiner als eine vorgegebene Toleranz $\widetilde{TOL} = c_{Newton} \cdot TOL$, $c_{Newton} < 1$, ist, wobei TOL die vom Benutzer geforderte relative Genauigkeit der Lösung beschreibt.

Der Startwert x^0, den wir über Extrapolation des Prädiktorpolynoms (2.9) erhalten, ist in der Regel nicht weit vom Lösungspunkt x^* entfernt und sollte somit im lokalen Konvergenzbereich des vereinfachten Newton-Verfahrens liegen. Die grundsätzliche Problemstellung ist hier also nicht, überhaupt eine Lösung zu finden (wie das zum Beispiel bei der Berechnung von konsistenten Anfangswerten der Fall ist), sondern den Lösungspunkt in jedem Integrationsschritt mit möglichst wenig Aufwand zu berechnen. Der Aufwand liegt zum einen in der Berechnung und der Zerlegung der Jacobi-Matrix, aber auch im Lösen des linearen Gleichungssystems (2.13) mit bereits zerlegter Jacobi-Matrix J. Um den Aufwand des letzten Teils relativ gering zu halten, soll die Konvergenzrate so gut sein, daß man nach maximal drei Newton-Iterationen den Lösungspunkt x^* mit der geforderten Genauigkeit erreicht hat.

Nach zwei Iterationen erhalten wir eine Schätzung für die Konvergenzrate

$$\delta_0 \approx \frac{\|\Delta x^1\|}{\|\Delta x^0\|}\,.$$

Ist $\delta_0 < \delta_{max}$, $\delta_{max} < 1$, zum Beispiel $\delta_{max} = 0.3$[1], so wird noch eine weitere Iteration durchgeführt. Ansonsten muß zunächst die schlechte Konvergenz behoben werden. Diese kann unterschiedliche Ursachen haben:

1. eine große Änderung der Koeffizienten α_i des BDF-Verfahrens oder eine große Änderung der Schrittweite h;
2. eine große Änderung der Einträge in den Matrizen $\frac{\partial f}{\partial x}$, $\frac{\partial A}{\partial x}$ oder $\frac{\partial g}{\partial x}$;
3. der Startwert $x^0 = x^P$ für das Newton-Verfahren ist zu schlecht.

In den ersten beiden Fällen ist der Wert κ zu groß, das heißt die Näherungsinverse $\tilde{J}^{-1}$ ist keine gute Approximation von J^{-1} mehr. Im letzten Fall ist ω zu groß, der Startwert liegt nicht im lokalen Konvergenzbereich des Newton-Verfahrens. Der BDF-Schritt muß mit verkürzter Schrittweite wiederholt werden.

[1] Der Wert $\delta_{max} = 0.3$ ergibt sich aus Formel (2.16) und wenn wir davon ausgehen, daß die Newton-Schritte eine Verbesserung um mindestens den Faktor red ($\|x^m - x^*\| \leq red\,\|x^0 - x^*\|$, $red = 1/12$) erreichen sollten. Von Schwerin hat in seiner Arbeit [vS97] eine allgemeine Tabelle aufgestellt für den Faktor δ_{max} in Abhängigkeit von der Anzahl der Iterationen m im Newton-Verfahren und dem Faktor red.

Im folgenden wird ein Algorithmus dargestellt, der nacheinander die schlechte Konvergenz beziehungsweise Divergenz überprüft und behebt:

1. Solange $\delta_0 < \delta_{max}$, wird die Zerlegung von $\tilde{J}$ aus dem vorherigen Schritt übernommen.

2. Falls die Konvergenzrate zu schlecht ist, wird $\tilde{J}$ neu berechnet mit aktuellen BDF-Koeffizienten und aktueller Schrittweite aber eingefrorenen Matrizen $\frac{\partial f}{\partial x}$, $\frac{\partial A}{\partial x}$ und $\frac{\partial g}{\partial x}$.

3. Ist die Konvergenzrate immer noch zu schlecht, so werden die Matrizen $\frac{\partial f}{\partial x}$ und $\frac{\partial g}{\partial x}$ und die Richtungsableitung $\frac{\partial A}{\partial x}\, v$, $v = \alpha_0 y_{n+1} + h\, c$, neu berechnet und J neu zerlegt.

4. Falls trotz allem innerhalb von drei Schritten keine Konvergenz erzielt werden konnte oder die Konvergenzrate nach zwei Schritten zu schlecht ist, so wird der Schritt mit verkleinerter Schrittweite wiederholt.

Bemerkung 2.3.1 (Monitor-Strategie)
In den meisten Codes wird der 2. Schritt ausgelassen und bei schlechten Konvergenzraten sofort die komplette Jacobi-Matrix J neu berechnet und zerlegt. Falls die Auswertung der Ableitungen der Modellfunktionen aufwendiger ist als die Zerlegung von J, so ist obige Strategie sehr effizient. Erfahrungen haben gezeigt, daß diese Strategie etwa ein Drittel bis die Hälfte an Auswertungen der Matrizen $\frac{\partial f}{\partial x}$, $\frac{\partial A}{\partial x}$ und $\frac{\partial g}{\partial x}$ einspart.

2.3.2 Fehlerschätzung

Schätzung des globalen Fehlers

Es bezeichne $y(t_{n+1})$ und $z(t_{n+1})$ die wahre Lösung der DAE zum Zeitpunkt t_{n+1}, y_{n+1} und z_{n+1} die vom BDF-Verfahren berechnete Lösung zum Zeitpunkt t_{n+1}. Der globale Fehler ist definiert durch die Differenz zwischen der approximierten Lösung und der wahren Lösung:

$$e^y_{n+1} = y_{n+1} - y(t_{n+1}), \quad e^z_{n+1} = z_{n+1} - z(t_{n+1})\,.$$

Betrachte die nichtlineare DAE zum Zeitpunkt t_{n+1}

$$\begin{aligned} A(t_{n+1}, y_{n+1}, z_{n+1})\, \dot{y}(t_{n+1}) &= f(t_{n+1}, y_{n+1}, z_{n+1}) \\ 0 &= g(t_{n+1}, y_{n+1}, z_{n+1})\,. \end{aligned}$$

Bezeichne ρy_{n+1} das Korrektorpolynom im Punkt t_{n+1}, das die Ableitung $h \cdot \dot{y}(t_{n+1})$ approximiert

$$\dot{y}(t_{n+1}) \approx -\frac{1}{h} \sum_{i=0}^{k} \alpha_i y_{n+1-i} =: -\frac{\rho y_{n+1}}{h}\,, \tag{2.17}$$

so erhält man

$$\begin{pmatrix} A(t_{n+1}, y_{n+1}, z_{n+1}) \cdot \frac{\rho y_{n+1}}{h} + f(t_{n+1}, y_{n+1}, z_{n+1}) \\ g(t_{n+1}, y_{n+1}, z_{n+1}) \end{pmatrix}$$

$$
\begin{aligned}
&= \left(\begin{array}{c} A(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \cdot \frac{\rho(y(t_{n+1}) + e^y_{n+1})}{h} \end{array} \right) \\
&\quad + \left(\begin{array}{c} f(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \\ g(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \end{array} \right) \\
&= \left(\begin{array}{c} A(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \cdot \left(-\dot{y}(t_{n+1}) + \tau_{n+1}(y) + \frac{\rho e^y_{n+1}}{h} \right) \\ 0 \end{array} \right) \\
&\quad + \left(\begin{array}{c} f(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \\ g(t_{n+1}, y(t_{n+1}) + e^y_{n+1}, z(t_{n+1}) + e^z_{n+1}) \end{array} \right) \\
&= \underbrace{\left(\begin{array}{c} -A(t_{n+1}, y(t_{n+1}), z(t_{n+1})) \cdot \dot{y}(t_{n+1}) + f(t_{n+1}, y(t_{n+1}), z(t_{n+1})) \\ g(t_{n+1}, y(t_{n+1}), z(t_{n+1})) \end{array} \right)}_{=0} \\
&\quad + \left(\begin{array}{c} A \cdot \left(\tau_{n+1}(y) + \frac{\rho e^y_{n+1}}{h} \right) + \left(A_y \, \frac{\rho y_{n+1}}{h} + f_y \right) e^y_{n+1} + \left(A_z \, \frac{\rho y_{n+1}}{h} + f_z \right) e^z_{n+1} \\ g_y \, e^y_{n+1} + g_z \, e^z_{n+1} \end{array} \right) \\
&\quad + \left\{ \text{Terme höherer Ordnung in } e^y_{n+1}, e^z_{n+1} \text{ und } \tau_{n+1} \right\}
\end{aligned}
$$

Funktionen ohne Argument bezeichnen dabei die Auswertung der Funktion an der Stelle $(t_{n+1}, y(t_{n+1}), z(t_{n+1}))$. Die Lösung $x_{n+1} = (y_{n+1}, z_{n+1})$ von (2.12) wird mit einem Newton-Verfahren berechnet. Durch den vorzeitigen Abbruch des Newton-Verfahrens entsteht ein weiterer Fehler im Verfahren. Wir bezeichnen im folgenden mit $\delta = (\delta_1^T, \delta_2^T)^T$ den Fehler durch den vorzeitigen Abbruch im Newton-Verfahren zusammen mit den Fehlertermen höherer Ordnung in e^y_{n+1}, e^z_{n+1} und τ_{n+1}.

Damit erhalten wir

$$
\left(\begin{array}{c} A \cdot \left(\tau_{n+1} + \frac{\rho e^y_{n+1}}{h} \right) + \left(A_y \, \frac{\rho y_{n+1}}{h} + f_y \right) e^y_{n+1} + \left(A_z \, \frac{\rho y_{n+1}}{h} + f_z \right) e^z_{n+1} \\ g_y \, e^y_{n+1} \; + \; g_z \, e^z_{n+1} \end{array} \right) = \left(\begin{array}{c} \delta_1 \\ \delta_2 \end{array} \right).
$$

Es ergibt sich folgende Approximation für den globalen Fehler:

$$
\left(\begin{array}{cc} \alpha_0 A + A_y \, \rho y_{n+1} + h \, f_y & A_z \, \rho y_{n+1} + h \, f_z \\ g_y & g_z \end{array} \right) \left(\begin{array}{c} e^y_{n+1} \\ e^z_{n+1} \end{array} \right) = \left(\begin{array}{c} h \, \delta_1 - A \left(h \, \tau_{n+1} - h \, c(e^y) \right) \\ \delta_2 \end{array} \right) \tag{2.18}
$$

Schätzung des lokalen Fehlers

Im folgenden beschreiben wir eine Fehlerschätzung und darauf aufbauend eine geeignete Schrittweitensteuerung, basierend auf den obigen Größen.

Da der globale Fehler nicht direkt berechnet werden kann ($c(e^y)$, δ_1 und δ_2 können nicht exakt berechnet werden), basiert die Fehlerkontrolle auf Schätzungen des lokalen Fehlers. Im folgenden leiten wir eine Schätzung des lokalen Fehlers, basierend auf dem lokalen Diskretisierungsfehler τ_{n+1} des Verfahrens, her (ähnliche Schätzungen des lokalen Fehlers

finden sich zum Beispiel auch bei Gear [Gea71], Lötstedt und Petzold [LP86, PL86] und Eich [Eic92]).

Der lokale Diskretisierungsfehler eines Verfahrens ist definiert durch die Differenz zwischen der exakten Lösung und dem Wert, der sich daraus ergibt, daß die exakte Lösung der Differentialgleichung in das Diskretisierungsschema – hier die BDF-Formeln – eingesetzt wird. Für ein BDF-Verfahren der Ordnung k ist der lokale Diskretisierungsfehler im Schritt $n+1$ also gegeben durch

$$\begin{aligned}
\tau_{n+1} &= \dot{x}(t_{n+1}) - \dot{x}_{n+1} \\
&= \dot{x}(t_{n+1}) + \frac{\rho x(t_{n+1})}{h} \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k} \frac{1}{\psi_j(n+1)} \tilde{\delta}_j(n+1) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k} \frac{1}{\psi_j(n+1)} \sum_{i=1}^{j} \tilde{\Phi}^*_i(n) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k} \sum_{i=j}^{k} \frac{1}{\psi_i(n+1)} \tilde{\Phi}^*_j(n) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k} \left(\sum_{i=1}^{j-1} \frac{1}{\psi_i(n+1)} - \sum_{i=1}^{k} \frac{1}{\psi_i(n+1)} \right) \tilde{\Phi}^*_j(n) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k+1} \left(\gamma_j(n+1) - \gamma_{k+1}(n+1) \right) \tilde{\Phi}^*_j(n) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1)\, x(t_{n+1}) + \sum_{j=1}^{k+1} \gamma_j(n+1) \tilde{\Phi}^*_j(n) - \gamma_{k+1}(n+1) \sum_{j=1}^{k+1} \tilde{\Phi}^*_j(n) \\
&= \dot{x}(t_{n+1}) - \gamma_{k+1}(n+1) x(t_{n+1}) + \dot{\tilde{P}}^P_{n+1}(t_{n+1}) - \gamma_{k+1}(n+1) \tilde{P}^P_{n+1}(t_{n+1}) \\
&= \dot{x}(t_{n+1}) - \dot{\tilde{P}}^P_{n+1}(t_{n+1}) - \gamma_{k+1}(n+1) \left(x(t_{n+1}) - \tilde{P}^P_{n+1}(t_{n+1}) \right).
\end{aligned}$$

Dabei bezeichnen die Tilde-Terme $\tilde{\Phi}^*_j(n)$ und $\tilde{P}^P_{n+1}$ die modifizierten dividierten Differenzen beziehungsweise das Prädiktorpolynom, die die exakte Lösung $x(t)$ an den Stellen $t_{n-k}, \ldots, t_n, t_{n+1}$ interpolieren beziehungsweise extrapolieren.

Mit

$$x(t) - \tilde{P}^P_{n+1}(t) = p_{k+1}(t) \nabla^{k+1}[x(t), x(t_n), \ldots, x(t_{n-k})]$$

und

$$\nabla^1[x(t), x(t)] := \lim_{\hat{t} \to t} \frac{x(\hat{t}) - x(t)}{\hat{t} - t} = \dot{x}(t)$$

erhalten wir für die Ableitung

$$\dot{x}(t) - \dot{\tilde{P}}^P_{n+1}(t) = \dot{p}_{k+1}(t) \nabla^{k+1}[x(t), x(t_n), \ldots, x(t_{n-k})]$$

$$
\begin{aligned}
&+p_{k+1}(t)\frac{d}{dt}\nabla^{k+1}[x(t),x(t_n),\ldots,x(t_{n-k})]\\
=\;&\dot{p}_{k+1}(t)\nabla^{k+1}[x(t),x(t_n),\ldots,x(t_{n-k})]\\
&+p_{k+1}(t)\nabla^{k+2}[x(t),x(t),x(t_n),\ldots,x(t_{n-k})]\,.
\end{aligned}
$$

Der lokale Diskretisierungsfehler ergibt sich somit als:

$$
\begin{aligned}
\tau_{n+1} =&\dot{x}(t_{n+1}) - \dot{\tilde{P}}^P_{n+1}(t_{n+1}) - \gamma_{k+1}(n+1)\,(x(t_{n+1}) - \tilde{P}^P_{n+1}(t_{n+1}))\\
=&\dot{p}_{k+1}(t_{n+1})\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&+ p_{k+1}(t_{n+1})\nabla^{k+2}[x(t_{n+1}),x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&- \gamma_{k+1}(n+1)p_{k+1}(t_{n+1})\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
=&\gamma_{k+2}(n+1)\cdot\psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&+ \psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+2}[x(t_{n+1}),x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&- \gamma_{k+1}(n+1)\cdot\psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
=&\frac{1}{\psi_{k+1}(n+1)}\cdot\psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&+ \psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+2}[x(t_{n+1}),x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
=&\frac{1}{\psi_{k+1}(n+1)}\cdot\tilde{\Phi}_{k+2}(n+1)\\
&+ \psi_1(n+1)\cdot\ldots\cdot\psi_{k+1}(n+1)\cdot\nabla^{k+2}[x(t_{n+1}),x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})].
\end{aligned}
\tag{2.19}
$$

Der globale Fehler setzt sich zusammen aus dem lokalen Diskretisierungsfehler des BDF-Verfahrens und aus dem Fehler, der daraus entsteht, daß nur eine genäherte Lösung interpoliert wird. Der erste Fehler ist dabei in der Regel der Hauptanteil.

Wir gehen davon aus, daß die zurückliegenden Werte mit hinreichender Genauigkeit berechnet wurden. Für Beispiele der Form $\dot{y} = -a\,y$, $a > 0$, kann der zweite Fehleranteil für BDF-Verfahren bis Ordnung 6 aufgrund der Stabilität und Konvergenz nicht akkumulieren.

Die Fehlerkontrolle in DAESOL basiert somit auf Schätzungen des lokalen Fehlers. Dabei wird das nichtäquidistante Gitter auch für die Fehlerschätzung und Ordnungs- und Schrittweitensteuerung berücksichtigt. Bezeichne $e^y_{\tau_{n+1}}$ und $e^z_{\tau_{n+1}}$ eine Approximation des lokalen Fehlers basierend auf dem lokalen Diskretisierungsfehler des BDF-Verfahrens. Gleichung (2.18) ergibt sich somit zu

$$
\begin{pmatrix} \alpha_0 A + A_y\,\rho y_{n+1} + h\,f_y & A_z\,\rho y_{n+1} + h\,f_z \\ g_y & g_z \end{pmatrix} \cdot \begin{pmatrix} e^y_{\tau_{n+1}} \\ e^z_{\tau_{n+1}} \end{pmatrix} = \begin{pmatrix} -h\,A\,\tau^y_{n+1} \\ 0 \end{pmatrix} \tag{2.20}
$$

Da es sehr aufwendig ist, den lokalen Fehler und eine neue Schrittweite aus obiger Formel zu berechnen, wählen wir eine vereinfachte Fehlerformel:

$$
\begin{aligned}
E_k(n+1) &= h_{n+1}\cdot\|\tau_{n+1}\|\\
&\doteq h_{n+1}\cdot\psi_1(n+1)\cdot\ldots\cdot\psi_k(n+1)\cdot\|\nabla^{k+1}x_{n+1}\|\,.
\end{aligned}
\tag{2.21}
$$

Nach jedem Schritt testen wir diese Fehlerformel. Falls der geschätzte Fehler $E_k(n+1)$ kleiner ist als eine bestimmte Toleranz TOL, wobei TOL die vom Benutzer vorgegebene relative Genauigkeit ist, so wird der aktuelle Schritt akzeptiert, ansonsten wird der Schritt zurückgewiesen und die Schrittweite reduziert. Die Schrittweitenreduktion nach einem zurückgewiesenen Schritt wird genauer am Ende dieses Kapitels beschrieben.

2.3.3 Schrittweiten- und Ordnungssteuerung

Zur Berechnung einer neuen Schrittweite und Ordnung für den nächsten Schritt übertragen wir die Fehlerformeln aus Schritt $n+1$ auf Schritt $n+2$ und leiten eine Schätzung für den Fehler im nächsten Schritt her, der nur auf Werten beruht, die im Schritt $n+1$ bekannt sind.

Mit Formel (2.19) als Schätzung für den lokalen Diskretisierungsfehler im Schritt $n+1$ erhalten wir für den akkumulierten Fehler im Schritt $n+2$

$$\begin{aligned}
\tau_{n+2} &= \psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\nabla^{k+1}[x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n+1-k})]\\
&\quad +\psi_1(n+2)\cdot\ldots\cdot\psi_{k+1}(n+2)\cdot\nabla^{k+2}[x(t_{n+2}),x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n+1-k})]\\
&= \psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\Big(\nabla^{k+1}[x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n+1-k})]\\
&\quad +\psi_{k+1}(n+2)\cdot\nabla^{k+2}[x(t_{n+2}),x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n+1-k})]\Big)\\
&= \psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\\
&\quad \Big(\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]+\psi_{k+2}(n+2)\cdot\nabla^{k+2}[x(t_{n+2}),\ldots,x(t_{n-k})]\\
&\quad +\psi_{k+1}(n+2)\cdot\nabla^{k+2}[x(t_{n+2}),\ldots,x(t_{n-k})]\\
&\quad +\psi_{k+1}(n+2)\cdot\psi_{k+2}(n+2)\cdot\nabla^{k+3}[x(t_{n+2}),x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n-k})]\Big)\\
&= \psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\Big(\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&\quad +(\psi_{k+1}(n+2)+\psi_{k+2}(n+2))\cdot\nabla^{k+2}[x(t_{n+1}),\ldots,x(t_{n-k-1})]\\
&\quad +(\psi_{k+1}(n+2)+\psi_{k+2}(n+2))\cdot\psi_{k+3}(n+2)\cdot\nabla^{k+3}[x(t_{n+2}),\ldots,x(t_{n-k-1})]\\
&\quad +\psi_{k+1}(n+2)\cdot\psi_{k+2}(n+2)\cdot\nabla^{k+3}[x(t_{n+2}),x(t_{n+2}),x(t_{n+1}),\ldots,x(t_{n-k})]\Big)\\
&\doteq \psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\Big(\nabla^{k+1}[x(t_{n+1}),x(t_n),\ldots,x(t_{n-k})]\\
&\quad +(\psi_{k+1}(n+2)+\psi_{k+2}(n+2))\cdot\nabla^{k+2}[x(t_{n+1}),\ldots,x(t_{n-k-1})]\Big).
\end{aligned}$$

Für den lokalen Fehler wählen wir eine Schätzung analog zu (2.21):

$$\begin{aligned}
E_k(n+2) &= h_{n+2}\cdot\|\tau_{n+2}\|\\
&\doteq h_{n+2}\cdot\psi_1(n+2)\cdot\ldots\cdot\psi_k(n+2)\cdot\|\nabla^{k+1}x_{n+1}\| \qquad (2.22)\\
&=: h_{n+2}^2\cdot q(h_{n+2})\cdot\|\nabla^{k+1}x_{n+1}\|
\end{aligned}$$

mit

$$q(h_{n+2}) = (h_{n+2}+\psi_1(n+1))\cdot\ldots\cdot(h_{n+2}+\psi_{k-1}(n+1))\,.$$

Die Schrittweite sollte so gewählt werden, daß im nächsten Schritt

$$E_k(n+2) \leq TOL$$

gilt. Da aus Fehlerformel (2.22) nicht einfach eine neue Schrittweite zu berechnen ist, leiten wir zunächst eine vereinfachte Fehlerformel her:

Die analoge Formel zu (2.21) auf äquidistantem Gitter ist

$$\hat{E}_k(n+2) := k!\, h^{k+1} \|\nabla^{k+1} x_{n+1}\| \,.$$

Mit $\hat{E}_k(n+2) \leq \widehat{TOL}$ erhält man eine erste Schätzung für die neue Schrittweite

$$\hat{h} = \sqrt[k+1]{\frac{\widehat{TOL}}{k!\, \|\nabla^{k+1} x_{n+1}\|}}, \tag{2.23}$$

die sogenannte *maximale gleichmäßige Schrittweite*, siehe Bleser [Ble86].

Um eine neue Schrittweite und Ordnung für den nächsten Schritt zu bestimmen, wird diese Formel für verschiedene Ordnungen $k' = k-1, k, k+1$ mit $\widehat{TOL} < TOL$, z. B. $\widehat{TOL} = \frac{1}{2} \cdot TOL$ ausgewertet. Wir verringern oder erhöhen die Ordnung um 1, falls die berechnete Schrittweite für die zugehörige Ordnung sich signifikant gegenüber der Ordnung im letzten Schritt geändert hat. Ansonsten wird die aktuelle Ordnung beibehalten.

Fehlerformel (2.23) scheint allerdings ungeeignet für die Berechnung im Fall variabler Schrittweiten. Zum einen basiert die Fehlerformel auf variablem Gitter (Faktor $\nabla^{k+1} x_{n+1}$), zum anderen wird die Schrittweite als konstant angenommen ($(k+1)$-te Wurzel).

Wir betrachten $\hat{h}$ als eine erste Näherung für die neue Schrittweite und testen, ob sie auch die genauere Fehlerformel (2.22) auf nichtäquidistantem Gitter erfüllt:

$$E_k(n+2) = h^2\, q(h)\, \|\nabla^{k+1} x_{n+1}\| \leq TOL. \tag{2.24}$$

Falls obige Ungleichung erfüllt ist, so wird die Schrittweite akzeptiert, ansonsten wird sie mit Hilfe von Fehlerformel (2.24) reduziert:

$$h^2 = \frac{1}{q(h^*)} \cdot \frac{TOL}{\|\nabla^{k+1} x_{n+1}\|}. \tag{2.25}$$

Dabei bezeichne h^* die zuvor gewählte Schrittweite.

Bemerkung 2.3.2 (Schrittweiten- und Ordnungssteuerung)
Shampine und Bogacki [SB89] haben gezeigt, daß eine Schrittweitensteuerung basierend auf Approximationen auf äquidistantem Gitter bei der Berechnung einer Schrittweite für den nächsten Schritt zu pessimistisch ist, wenn eine Vergrößerung der Schrittweite angestrebt wird, andererseits bei Schrittzurückweisungen die Verkleinerung oft zu optimistisch gewählt wird. Die in DAESOL verwendeten Schrittweitenschätzungen gehen zunächst von Formeln auf äquidistantem Gitter aus, der sogenannten maximalen gleichmäßigen Schrittweite, *siehe Bleser [Ble86]. Sie werden aber im Anschluß mit Hilfe der Formeln auf variablem Gitter korrigiert. Gleichzeitig wird zugelassen, daß sich die Schrittweite in jedem Schritt ändern kann.*

Auch die Ordnung wird relativ schnell hochgeschaltet. Dabei wird das von Gear und Watanabe [GW74] aus Stabilitätsgründen geforderte Festhalten der Ordnung für ein bis drei Schritte (abhängig von der aktuellen Ordnung) bei einer Erhöhung der Ordnung um eins in der Praxis immer eingehalten.

Bleser [Ble86] hat für die Beispiele aus STIFF DETEST, einer Beispielsammlung von steifen gewöhnlichen Differentialgleichungssystemen von Enright et al. [EHL75], gezeigt, daß eine Schrittweiten- und Ordnungssteuerung, die auf den Formeln auf nichtäquidistantem Gitter beruht, zu einem schnelleren Hochschalten der Ordnung und zu weniger zurückgewiesenen Schritten führt als bei konventionellen Schätzungen.

Bemerkung 2.3.3 (Schrittweitensteuerung und Monitor-Strategie)
In vielen Codes wird die Schrittweite über mehrere Schritte hinweg konstant gehalten, teilweise auch, um eine neue Auswertung und Zerlegung der Iterationsmatrix J zu vermeiden. Falls die Schrittweite dann verändert wird, so meist relativ stark, was allerdings leicht zu Stabilitätsproblemen führen kann. Zum anderen kann es leicht vorkommen, daß die Iterationsmatrix $\tilde{J}$ im Newton-Verfahren, die auch explizit von der Schrittweite abhängt, nach großen Schrittweitenänderungen keine gute Näherung mehr für J ist. Die Konvergenzraten im Newton-Verfahren werden schlecht und die Iterationsmatrix muß mit aktueller Schrittweite neu zerlegt werden. Da in den meisten Codes der 2. Schritt aus der Monitor-Strategie fehlt, erfordert dies gleichzeitig eine Auswertung der Ableitungen der Modellfunktionen.

Die oben beschriebene Schrittweitensteuerung führt auf eine gleichmäßige Änderung der Schrittweiten. Wenn sich die Schrittweite nach einigen Schritten zu sehr geändert hat, so wird zunächst nur die Iterationsmatrix J mit aktueller Schrittweite und BDF-Koeffizienten neu zerlegt.

Schrittweitensteuerung nach Schrittzurückweisungen

Schrittzurückweisungen können aus zwei Gründen auftreten:

- zum einen, weil der geschätzte Fehler zu groß ist
- und zum anderen, weil das Newton-Verfahren nicht innerhalb von drei Schritten konvergiert hat.

Hat das Newton-Verfahren konvergiert, aber der geschätzte Fehler ist zu groß, so berechnen wir eine neue Schrittweite aus Fehlerformel (2.21)

$$h_{n+1}^{(neu)} = \frac{h_{n+1}^{(alt)} \cdot TOL'}{E_k(n+1)}, \tag{2.26}$$

mit $TOL' = c_{red} \cdot TOL$, $c_{red} \leq 1$. Diese Formel beruht wiederum auf Approximationen basierend auf dem variablen Gitter.

Falls das Newton-Verfahren nicht konvergiert hat, so ist $\|\Delta x^0\|$ eine Approximation für

$$\|x_{n+1}^C - x_{n+1}^P\| = \psi_1(n+1) \cdot \ldots \cdot \psi_k(n+1) \cdot \|\nabla^{k+1} x_{n+1}\|$$

und wir erhalten eine Schätzung für den lokalen Fehler mit

$$\|E_k\| = \frac{h}{\psi_{k+1}(n+1)} \cdot \|\Delta x^{(0)}\| \,. \tag{2.27}$$

Die Schrittweite sollte so weit verkleinert werden, daß das Newton-Verfahren innerhalb von zwei Schritten konvergiert (zur Sicherheit wird, falls nötig, noch eine weitere Iteration erlaubt).

Das heißt für die Konvergenzrate

$$\delta_0^{(neu)} \overset{!}{\leq} \frac{1}{4} \,.$$

Da die oben dargestellten Schritte 1. - 4. der Monitor-Strategie nacheinander ausgeführt wurden, wurde die Zerlegung von $\tilde{J}$ mit aktuellen Matrizen $\frac{\partial f}{\partial x}$, $\frac{\partial A}{\partial x}$ und $\frac{\partial g}{\partial x}$, aktuellen BDF-Koeffizienten und aktueller Schrittweite durchgeführt. Die Größe κ in Fromel (2.15) ist dann gleich Null und wir erhalten

$$\delta_0 = \frac{\omega^0}{2} \|\Delta x^0\| \,.$$

Mit

$$\delta_0^{(neu)} = \frac{\omega^0}{2} \|\Delta x_{(neu)}^0\| \overset{!}{\leq} \frac{1}{4}$$

haben wir somit

$$\|\Delta x_{(neu)}^0\| \leq \frac{\|\Delta x^0\|}{4 \cdot \delta_0} \,.$$

Die Fehlerformel aus (2.21) für die neue Schrittweite sollte dann $E_k(n+1) \leq \widehat{TOL}$ an Stelle von $E_k(n+1) \leq TOL$ erfüllen mit verschärfter Toleranz

$$\widehat{TOL} = \frac{h}{\psi_{k+1}(n+1)} \cdot \frac{\|\Delta x^0\|}{4 \cdot \delta_0} \,.$$

Wir erhalten eine neue Schrittweite $h_{n+1}^{(neu)}$ aus Formel (2.26) mit $\widehat{TOL}$ an Stelle von TOL'.

2.3.4 Skalierung

Bei der Fehlerschätzung und der Schrittweiten- und Ordnungssteuerung tritt immer wieder die Norm $\|\cdot\|$ auf. Da die Komponenten der Lösung oft von ganz unterschiedlicher Größenordnung sind, verwenden wir hierfür eine gewichtete Norm an Stelle einer l_2-Norm $\|x\|_2$

$$\|x_{n+1}\| = \frac{1}{n_x} \sqrt{\sum_{i=1}^{n_x} \left(\frac{x_{n+1,i}}{xscal_{n+1,i}} \right)^2} \,.$$

$x_{n+1,i}$ beschreibt dabei die i-te Komponente der approximierten Lösungstrajektorie im Schritt $n+1$ und $xscal_{n+1,i}$ den zugehörigen Skalierungsfaktor.

In DAESOL kann der Benutzer unter den folgenden beiden Skalierungen wählen:

a) Skalierung wie sie in den meisten Integratoren verwendet wird (siehe etwa Brenan et al. [BCP96]):

$$xscal_{n+1,i} = |x_{n+1,i}| + ATOL_i/TOL, \quad i = 1, \ldots, n_x \tag{2.28}$$

b) Skalierung, die zusätzlich die Größenordnung der Komponenten zu früheren Zeitpunkten berücksichtigt:

$$xscal_{n+1,i} = max(|x_{n+1,i}|, xscal_{n,i}, ATOL_i), \quad i = 1, \ldots, n_x \tag{2.29}$$

mit $xscal_{0,i} = 0$, TOL die relative und $ATOL$ die absolute Genauigkeit, die vom Benutzer vorgegeben werden.

Der Parameter $ATOL$ kann dabei als skalare Größe gewählt werden ($ATOL_i = ATOL_1$, $i = 2, \ldots, n_x$) oder als Vektor ($ATOL \in I\!R^{n_x}$). Letzteres ist dann sinnvoll, wenn die Lösungskomponenten sehr unterschiedlich groß sind. Im Fall *a*) hat $ATOL$ die Bedeutung des absoluten Fehlers. Die Größe $(-\log_{10}(TOL))$ gibt die Anzahl an signifikanten Stellen für die Lösungstrajektorie an. Werte kleiner als $ATOL_i$ gehen nicht mehr signifikant in die Lösung ein. Im Modus *b*) hat die Variable $ATOL$ die Bedeutung eines Skalierungsfaktors. Sie beschreibt die Größenordnung, für die wir noch die Genauigkeit von $(-\log_{10}(TOL))$ Ziffern erhalten möchten. Sind die Werte kleiner als $ATOL_i$, so muß die Komponente nur noch für die $-\log_{10}(TOL) + \log_{10}(ATOL_i)$ Ziffern genau berechnet werden.

Im Fall *b*) hängt die aktuelle Skalierung von dem Wert aus dem vorherigen Schritt ab. Die Idee ist, den maximalen Wert der einzelnen Komponenten in der Fehlerschätzung mit zu berücksichtigen. In manchen Fällen, wenn eine Komponente im Laufe der Integration stark abfällt, die kleinen Werte aber trotzdem signifikant in die Berechnungen eingehen, ist der Skalierungsmodus *a*) zu empfehlen.

Kapitel 3

Strategien in der Startphase

Bei der numerischen Integration von DAE-Systemen, insbesondere mit Mehrschrittverfahren, muß gerade der Startphase besondere Beachtung geschenkt werden. Bei DAE-Systemen müssen die Anfangswerte konsistent sein (siehe auch Abschnitt 1.3). Sind die algebraischen Gleichungen hochgradig nichtlinear, so können konsistente Anfangswerte oft nicht mehr mit einem Newton-Verfahren berechnet werden. Im Programmpaket DAESOL stehen zur konsistenten Initialisierung zwei Möglichkeiten zur Verfügung: Zum einen können die konsistenten Anfangswerte mit einem Homotopie-Verfahren berechnet werden. Der Homotopieweg kann dabei eine vorgegebene Standardhomotopie oder ein vom Benutzer definierter, physikalisch fundierter Weg sein. Im Optimierungskontext ist eine Relaxierung der algebraischen Gleichungen zu empfehlen. Diese garantiert Konsistenz bei der Lösung des Anfangswertproblems in jeder Iteration des Optimierungsproblems.

Ein weiteres Problem ist die Generierung von zurückliegenden Werten bei Mehrschrittverfahren. Dies kann entweder mit einem Einschrittverfahren höherer Ordnung erfolgen oder durch Starten des Mehrschrittverfahrens mit Ordnung 1 und anschließendem langsamem Hochschalten der Ordnung. In letzterem Fall ist die Approximation bei niedriger Ordnung weniger genau, weshalb in der Regel eine kürzere Schrittweite im Vergleich zu höheren Ordnungen nötig ist.

Im folgenden beschreiben wir zunächst ein Homotopie-Verfahren zur konsistenten Initialisierung bei DAE-Systemen vom Index 1 wie es im Programmpaket DAESOL implementiert ist. Wie bei der Fehlerschätzung und Schrittweitensteuerung für das BDF-Verfahren in DAESOL beruht die Schrittweitensteuerung auf Fehlerformeln auf dem tatsächlichen nichtäquidistanten Gitter; für die Lösung der Lineare-Algebra-Teilprobleme wird eine Monitor-Strategie ähnlich zu 2.3.1 angewendet.

Im Optimierungskontext ist eine Relaxierung der algebraischen Gleichungen sinnvoll, wie sie im Anschluß dargestellt wird. Die dabei auftretenden Strukturen werden in DAESOL ausgenutzt.

Zur Generierung von Startdaten wurde ein Runge-Kutta-Verfahren der Ordnung 4 implementiert. Da auch 4 der 7 internen Stufen von der Ordnung 3 sind, reicht ein Schritt

mit dem Einschrittverfahren aus, um alle nötigen zurückliegenden Werte für das BDF-Verfahren zu generieren. Das implementierte Runge-Kutta-Verfahren ist ein SDIRK-Verfahren, so daß für die Lösung der Lineare-Algebra-Teilprobleme jeweils die gleiche zerlegte Iterationsmatrix verwendet werden kann. Dabei wurde auch berücksichtigt, daß diese im Anschluß für das BDF-Verfahren wiederverwendet werden kann. Die Fehlerschätzung für das SDIRK-Verfahren ist genau auf diejenige des BDF-Verfahrens zugeschnitten.

3.1 Konsistente Initialisierung

Zur Lösung eines Anfangswertproblems für eine DAE der Form

$$F(t, y, \dot{y}) = 0, \quad y(t_0) = y_0, \dot{y}(t_0) = \dot{y}_0,$$

müssen die Anfangswerte konsistent sein, d.h. $F(t_0, y_0, \dot{y}_0) = 0$ (siehe Abschnitt 1.3).

Würde man die Integration mit inkonsistenten Anfangswerten starten, so würde das Verfahren (falls das Newton-Verfahren zur Lösung der impliziten nichtlinearen Gleichungen überhaupt konvergiert) nach einem Schritt einen konsistenten Punkt, d.h. $F(t_1, y_1, \dot{y}_1) = 0$ berechnen. Probleme ergäben sich dann bei der Fehlerschätzung, da der Fehler im Anfangsschritt mit $h_0 \to 0$ in der Regel konstant ist oder zumindest nicht mit $h_0 \to 0$ gegen 0 geht, da er zusätzlich den Abstand zwischen inkonsistentem und konsistentem Anfangswert enthält. Die Fehlerschätzung würde auch bei Reduzierung der Schrittweite keinen berechneten Wert akzeptieren.

Nachfolgend beschränken wir uns wiederum auf DAEs vom Index 1 der Form (2.11). Hierbei sind die differentiellen Variablen y die freien Variablen. Die algebraischen Variablen z können durch die algebraischen Gleichungen eindeutig bestimmt werden, da g_z regulär ist (Index 1-Bedingung). Um die konsistenten Anfangswerte für die algebraischen Variablen zu bestimmen, muß man die Nullstelle von $g(t_0, y_0, z)$ berechnen.

Ein Vollschritt-Newton-Verfahren zur Berechnung der Nullstelle von $g(t_0, y_0, z)$ konvergiert nur dann, wenn die Startwerte für $z(t_0)$ im lokalen Konvergenzbereich des Verfahrens liegen. Für hochgradig nichtlineare Systeme kann der Konvergenzbereich des Vollschritt-Newton-Verfahrens sehr klein sein. Manche Komponenten von z sind oft gänzlich unbekannt, da sie unter anderem keinen Messungen zugänglich sind. Es ist dann schwierig oder praktisch unmöglich, geeignete Startschätzungen für die einzelnen Komponenten anzugeben.

Zur Berechnung konsistenter Anfangswerte sind dann Verfahren mit Globalisierungsstrategien nötig, wie etwa globalisierte Newton-Verfahren oder Homotopie-Verfahren.

Für das Programmpaket DAESOL wurde ein Homotopie-Verfahren entwickelt und implementiert, das konsistente Anfangswerte berechnet, falls die Lösung mit einem Vollschritt-Newton-Verfahren mit den gegebenen Startschätzungen nicht gefunden werden kann.

3.1.1 Homotopie-Verfahren

Zur Bestimmung der Nullstelle einer Funktion $F : \mathbb{R}^{n_z} \to \mathbb{R}^{n_z}$ wird das Nullstellenproblem $F(z) = 0$ in eine Schar von Nullstellenproblemen $H(z, \alpha) = 0$, $H : \mathbb{R}^{n_z} \times \mathbb{R} \to \mathbb{R}^{n_z}$, eingebettet, etwa

$$H_1(z, \alpha) = F(z) - (1 - \alpha)F(z_0) \tag{3.1a}$$

$$\text{oder} \quad H_2(z, \alpha) = \alpha F(z) + (1 - \alpha)(z - z_0)\,, \tag{3.1b}$$

für die zum Zeitpunkt $\alpha = 0$ die Nullstelle von $H(z, \alpha)$ leicht zu ermitteln ist (hier z. B. $H_i(z_0, 0) = 0$, $i = 1, 2$).

Bemerkung 3.1.1 (Homotopieweg)
Im Programmpaket DAESOL stehen dem Benutzer die beiden Standard-Homotopien (3.1a) und (3.1b) zur Verfügung. Bei vielen Anwendungen ist es allerdings sinnvoller und eventuell sogar notwendig, eine Einbettung des Nullstellenproblems in einen physikalisch relevanten Homotopieweg vorzunehmen. Der Homotopieparameter α entspricht dann oft einer physikalisch interpretierbaren Größe. Der Benutzer hat die Möglichkeit, einen solchen Homotopieweg selbst bereitzustellen.

Wir lösen sukzessive eine Folge von Nullstellenproblemen

$$H(z, \alpha_{n+1}) = 0, \quad 0 = \alpha_0 < \alpha_1 < \cdots < \alpha_{N_H} = 1\,.$$

Die Lösung von

$$H(z_{n+1}, \alpha_{n+1}) = 0 \tag{3.2}$$

berechnen wir mit einem vereinfachten Newton-Verfahren. Dabei sollten der Startwert $z_{n+1}^{(0)}$ und α_{n+1} so gewählt werden, daß $z_{n+1}^{(0)}$ im lokalen Konvergenzbereich des Newton-Verfahrens liegt. Als Startwert könnte man die Lösung z_n von $H(z_n, \alpha_n) = 0$ verwenden.

Einen genaueren Startwert erhält man, indem man ein Polynom durch bereits berechnete Werte von z an früheren Gitterpunkten legt und zum Zeitpunkt α_{n+1} auswertet. Das Extrapolationspolynom durch die zurückliegenden Werte $z_n, \ldots, z_{n-k}$ kann analog zum Prädiktorpolynom des BDF-Verfahrens mit Hilfe der modifizierten dividierten Differenzen berechnet werden

$$P_{n+1}(\alpha_{n+1}) = \sum_{n=0}^{k} \prod_{j=1}^{n} (\alpha_{n+1} - \alpha_{n+1-j}) \nabla^j z_n.$$

Die Schrittweite $h_{n+2} = \alpha_{n+2} - \alpha_{n+1}$ für den nächsten Schritt wird so gewählt, daß der zugehörige Startwert $z_{n+2}^{(0)} = P_{n+2}(\alpha_{n+2})$ im lokalen Konvergenzbereich des Newton-Verfahrens liegt.

Bezeichne $z_{n+2}^P = P_{n+2}(\alpha_{n+2})$ den Prädiktor für z im Schritt $n+2$ und $z(\alpha_{n+2})$ die exakte Lösung, d.h. $H(z(\alpha_{n+2}), \alpha_{n+2}) = 0$.

Der Fehler des Startwerts $z^{(0)}_{n+2} = z^P_{n+2}$ ist von der Form

$$\begin{aligned}\|z^P_{n+2} - z(\alpha_{n+2})\| &\approx (\alpha_{n+2} - \alpha_{n+1}) \cdot \ldots \cdot (\alpha_{n+2} - \alpha_{n+1-k}) \cdot \|\nabla^{k+1} z_{n+2}\| \\ &\doteq (\alpha_{n+2} - \alpha_{n+1}) \cdot \ldots \cdot (\alpha_{n+2} - \alpha_{n+1-k}) && =: W(\alpha_{n+2}) \\ &\quad \cdot \|\nabla^{k+1} z_{n+1}\| && =: \eta\end{aligned}$$

η approximiert dabei die $(k+1)$-te Ableitung von z

$$\nabla^{k+1} z_{n+1} = \nabla^{k+1} z_{n+2} - (\alpha_{n+2} - \alpha_{n-k})\, \nabla^{k+2} z_{n+2} \doteq \nabla^{k+1} z_{n+2}$$

und somit

$$\eta := \|\nabla^{k+1} z_{n+1}\| \approx \|\nabla^{k+1} z(\alpha_{n+2})\| = \frac{\|z^{(k+1)}(\xi)\|}{(k+1)!}, \quad \xi \in [\alpha_{n-k}, \ldots, \alpha_{n+2}].$$

Für die Lösung des Nullstellenproblems (3.2) verwenden wir analog zur Lösung der nichtlinearen Gleichungssysteme im BDF-Verfahren eine Monitor-Strategie (siehe Abschnitt 2.3.1). Dabei soll die Schrittweite für den nächsten Schritt so gewählt werden, daß das vereinfachte Newton-Verfahren innerhalb von zwei Iterationen als konvergent angesehen werden kann, das heißt, daß der Fehler zwischen der zweiten Iterierten z^2 und dem Lösungspunkt z^* kleiner als eine vorgegebene Toleranz TOL_{Hom} ist. Zur Bestimmung einer geeigneten Schrittweite für den Homotopie-Parameter α betrachten wir den lokalen Kontraktionssatz für Homotopie-Verfahren (siehe Bock [Boc87] für Homotopie-Verfahren bei Parameterschätzproblemen), hier in etwas abgewandelter Form für ein vereinfachtes Newton-Verfahren:

Satz 3.1.1 (Lokaler Kontraktionssatz für Homotopie-Verfahren)
Sei $J = \frac{\partial H}{\partial z}$ die Jacobi-Matrix von H und $\tilde{J}^{-1}$ bezeichne eine Näherungsinverse von J. Für alle $\tau \in [0,1]$ und alle m gibt es dann Werte ω und κ, so daß

$$\|\tilde{J}^{-1}(J(z^m) - J(z^m - \tau \Delta z^m)) \cdot \Delta z^m\| \le \omega^m \tau \|\Delta z^m\|^2, \quad \omega^m \le \omega < \infty$$

$$\|\tilde{J}^{-1}(F(z^m) - J(z^m)\tilde{J}^{-1} F(z^m))\| \le \kappa^m \|\Delta z^m\|, \quad \kappa^m \le \kappa < 1$$

und für ein δ mit $\kappa < \delta < 1$ sei

$$W(\alpha_{n+2}) \le \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega \eta}, \quad \gamma := \kappa + 1. \tag{3.3}$$

Dann gelten die folgenden Abschätzungen

$$\begin{aligned}\|z^0 - z^*\| &\le \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega} \\ \|\Delta z^0\| &\le \frac{2(\delta - \kappa)}{\omega}\end{aligned}$$

und für die m-te Iterierte gilt die a priori Abschätzung

$$\|z^m - z^*\| \le \|\Delta z^0\| \frac{\delta^m}{1 - \delta}. \tag{3.4}$$

Beweis: siehe Bock [Boc87].

Bemerkung 3.1.2

a) *Falls $z(\alpha)$ als Funktion auf $[0,1]$ existiert und gleichmäßig stetig ist und $\omega(\alpha)$ für alle $\alpha \in [0,1]$ beschränkt ist (und gegebenenfalls $\kappa(\alpha) < \hat{\kappa} < 1$ falls die Näherungsinverse $\tilde{J}^{-1}$ ungleich der Inversen J^{-1} von H ist), dann existiert eine endliche Folge der α_i, so daß das Newton-Verfahren immer konvergiert.*
(Beweis: Lokale Existenz von $z(\alpha)$ aus Satz über implizite Funktionen).

b) *Das Verfahren kann zusammenbrechen, falls H_z für ein $\alpha \in [0,1]$ singulär ist.*

Betrachtet man den Kontraktionssatz für das vereinfachte Newton-Verfahren, so sieht man, daß insbesondere

$$W(\alpha_{n+2}) = (\alpha_{n+2} - \alpha_{n+1}) \cdot \ldots \cdot (\alpha_{n+2} - \alpha_{n+1-k}) \leq \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega\eta} \tag{3.5}$$

zur Sicherung der Konvergenz des Verfahrens gelten muß. Wir benötigen hierfür Schätzungen für κ und ω:

$$\begin{aligned} \kappa &\approx \frac{\|\tilde{J}^{-1} \cdot (H(z^0) + J(z^0)\Delta z^0)\|}{\|\Delta z^0\|} \\ \omega &\approx 2 \cdot \frac{\|\tilde{J}^{-1} \cdot (H(z^1) - H(z^0) - J(z^0)\Delta z^0)\|}{\|\Delta z^0\|^2}. \end{aligned}$$

κ dient als ein Maß für die Güte der Näherungsinversen $\tilde{J}^{-1}$ und ω für die Nichtlinearität der Jacobi-Matrix J. ($\frac{1}{\omega}$ beschreibt den Radius des Gebietes um z, in dem die Linearisierung $H(\tilde{z}) + J(\tilde{z})(z - \tilde{z})$ eine gute Näherung für $H(z)$ ist.)

Um die Größen κ and ω zu bestimmen, benötigen wir zur Berechnung der Richtungsableitung

$$J(z^0)\Delta z^0 = \lim_{\epsilon \to 0} \frac{H(z^0 + \epsilon \cdot \Delta z^0) - H(z^0)}{\epsilon}$$

nur eine zusätzliche Auswertung von H, außerdem noch eine Auswertung des linearen Gleichungssystems $\tilde{J}\, z = H(z^0) + J(z^0)\, \Delta z^0$ mit zerlegter Matrix $\tilde{J}$, jedoch keine Berechnung oder Zerlegung von J beziehungsweise $\tilde{J}$. Der Term $\tilde{J}^{-1}\, H(z^1)$ ist gleich Δz^1 und wurde bereits im Newton-ähnlichen Verfahren berechnet.

Zur Bestimmung einer Schrittweite, die Ungleichung (3.5) erfüllt, mit

$$\begin{aligned} W(\alpha_{n+2}) &= h_{n+2} \cdot (h_{n+2} + \alpha_{n+1} - \alpha_n) \cdot \ldots \cdot (h_{n+2} + \alpha_{n+1} - \alpha_{n+1-k}) \\ &=: h_{n+2} + q(h_{n+2}) \end{aligned}$$

beschränken wir uns zunächst wiederum auf Formeln auf äquidistantem Gitter

$$h^{k+1}(k+1)! \leq \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega\eta}.$$

Eine erste Schätzung für die neue Schrittweite erhalten wir mit

$$h = \sqrt[k'+1]{\frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{(k+1)!\omega\|\nabla^{k+1} z_{n+1}\|}}, \quad k' = k-1, k, k+1.$$

Wir werten obige Formel für die aktuelle Ordnung k und eine Ordnung höher beziehungsweise niedriger aus und wählen jene Ordnung aus, die die größtmögliche Schrittweite (maximale gleichmäßige Schrittweite, analog zur Schrittweitensteuerung im BDF-Verfahren) liefert.

Damit erhalten wir eine erste Schätzung für die neue Schrittweite. Wir prüfen, ob die so erhaltene Schrittweite ebenso die Fehlerformel auf variablem Gitter erfüllt:

$$h \cdot q(h) \leq \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega \cdot \|\nabla^{k+1} z_{n+1}\|}. \tag{3.6}$$

Wenn ja, so wird die neue Schrittweite akzeptiert, andernfalls reduzieren wir die Schrittweite nach Formel (3.6)

$$h^{(new)} = \frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega \cdot q(h) \cdot \|\nabla^{k+1} z_{n+1}\|}.$$

Um den Lineare-Algebra-Aufwand niedrig zu halten, soll das vereinfachte Newton-Verfahren innerhalb von zwei Iterationen konvergieren. Das Inkrement in der zweiten Newton-Iteration sollte also von der gleichen Größenordnung sein wie eine vorgegebene Fehlertoleranz TOL_{Hom}:

$$\|z_{n+2}^P - z^*\| = W(\alpha_{n+2}) \cdot \|\nabla^{k+1} z_{n+1}\| \leq \frac{TOL_{Hom}}{\delta^2},$$

δ beschreibt dabei eine maximal zugelassene Konvergenzrate des Newton-Verfahrens (hier z. B. $\delta = 1/4$). Somit muß die Schrittweite gleichzeitig

$$h \leq \frac{TOL_{Hom}}{\delta^2\, q(h)\, \|\nabla^{k+1} z_{n+1}\|}$$

erfüllen, woraus insgesamt folgt

$$\begin{aligned} h &\leq \frac{M}{q(h)\, \|\nabla^{k+1} z_{n+1}\|} \\ \text{mit } M &= \min\left(\frac{\sqrt{\gamma^2 + 4(\delta - \kappa)} - \gamma}{\omega}, \frac{TOL_{Hom}}{\delta^2} \right). \end{aligned}$$

3.1.2 Relaxierte Formulierung der algebraischen Gleichungen

Die numerische Lösung von Optimierungsproblemen mit einem DAE-System als Nebenbedingung erfordert in der Regel die mehrmalige Lösung eines Anfangswertproblems der Form (2.11) mit unterschiedlichen Anfangswerten und gegebenenfalls unterschiedlichen Werten der Optimierungsvariablen. Wird das Optimierungsproblem mit einem Verfahren vom Newton-Typ gelöst (zum Beispiel Gauß-Newton- oder SQP-Verfahren), so ist die Konsistenzbedingung in der Regel nach jeder Newton-Iteration verletzt. Dies gilt insbesondere dann, wenn die algebraischen Gleichungen von den zu optimierenden Variablen abhängen. Eine Relaxierung der algebraischen Gleichungen zur Lösung des Anfangswertproblems ist dann ratsam (siehe Bock et al. [BES88]). Dies führt auf ein Anfangswertproblem der Form

$$\begin{aligned} A(t,y,z)\,\dot{y} &= f(t,y,z) \\ 0 &= g(t,y,z) - \vartheta(t)\,g(t_0,y_0,z_0), \\ y(t_0) &= y_0,\ z(t_0) = z_0. \end{aligned} \tag{3.7}$$

Die Funktion $\vartheta : \mathbb{R} \to \mathbb{R}$ sei dabei stetig und hinreichend oft differenzierbar. Sie kann entweder identisch 1 sein, was einer Integration auf einer anderen Niveauebene entspricht, oder sie ist monoton fallend mit $\vartheta(t_0) = 1$ und $\vartheta(t) \geq 0$.

Die relaxierte Formulierung sichert konsistente Anfangswerte, auch wenn die Konsistenzbedingung für das ursprüngliche DAE-System verletzt ist.

Nimmt man die Konsistenzbedingungen

$$g(t_0, x_0) = 0 \tag{3.8}$$

als zusätzliche Gleichungsnebenbedingungen in das Optimierungsproblem auf, so ist die Konsistenz des ursprünglichen DAE-Systems im Lösungspunkt des Optimierungsproblems garantiert.

Bemerkung 3.1.3 (Wahl der Relaxierung)
Leineweber [Lei99] hat für komplexe Optimierungsprobleme bei Destillationskolonnen gezeigt, daß eine Funktion

$$\vartheta(t) = \exp\left(-\theta\,\left(\frac{t-t_0}{t_{end}-t_0}\right)\right) \tag{3.9}$$

mit $\theta \in [5, 10]$ für Optimierungsprobleme im Durchschnitt zu einer geringeren Anzahl an Newton-Iterationen für das Optimierungsproblem und zu einer geringeren Anzahl an zurückgewiesenen Schritten im BDF-Verfahren führt als für Werte, die deutlich kleiner oder größer sind.

Bemerkung 3.1.4 (Alternative Berechnung konsistenter Anfangswerte)
Eine Alternative zur Berechnung konsistenter Anfangswerte ist, den Integrator mit $f \equiv 0$ und einer relaxierten Formulierung der algebraischen Gleichungen der Form (3.7) zu starten. Dies ist äquivalent zum Homotopiepfad H_1 (3.1a) für

$$\vartheta(t) = \exp\left(-\theta\,\left(\frac{t-t_0}{t_{end}-t_0}\right)\right)$$

$\theta \geq 0$ geeignet, und $\vartheta(t) = 1 - \alpha$, α der Homotopieparameter aus dem vorhergehenden Abschnitt und falls $\frac{\partial g}{\partial t} \equiv 0$.

Der Unterschied zwischen BDF- und Homotopie-Verfahren liegt in der unterschiedlichen Schrittweitensteuerung. Die Schrittweite im BDF-Verfahren wird so gewählt, daß der aktuelle Wert z_{n+1} an der Stelle t_{n+1} die Fehlerformel für den Korrektor erfüllt. Angewandt auf den Homotopiepfad H wird dabei die Genauigkeit der Lösung während des Homotopiewegs kontrolliert. In der Regel ist man aber nur an dem Punkt z mit $H(z, 1) = 0$ interessiert. Die Wahl der Schrittweite aufgrund der Konvergenzeigenschaften des Newton-Verfahrens spielt bei der Berechnung einer neuen Schrittweite im BDF-Verfahren nur eine untergeordnete Rolle. Im Homotopie-Verfahren wird die Schrittweite genau so gewählt, daß das Newton-Verfahren in maximal zwei Schritten konvergiert. Dies wird im BDF-Verfahren zur Bestimmung einer neuen Schrittweite nur dann berücksichtigt, wenn der Schritt bereits zurückgewiesen wurde, weil das Newton-Verfahren nicht konvergiert hat.

3.2 Runge-Kutta-Starter für BDF-Verfahren

Bei der Simulation von dynamischen Systemen mit Mehrschrittverfahren kann die Startphase einen erheblichen Teil an der Gesamtrechenzeit benötigen. Die Integration beginnt mit Ordnung 1 und kleinen Schritten. Innerhalb der nächsten Schritte wird die Ordnung langsam hochgeschaltet. Die Schrittweite vergrößert sich in der Regel allmählich, bis das Mehrschrittverfahren bei der für das Modell geeigneten Ordnung und Schrittweite angelangt ist. Dabei wird vermehrt Rechenzeit benötigt, da zum einen zunächst nur mit kleinen Schritten gerechnet wird, zum anderen sich die Schrittweite im Laufe der Integration vergrößert und somit eine neue Zerlegung der Iterationsmatrix für das Newton-Verfahren notwendig ist. Außerdem sind die berechneten Werte für niedrigere Ordnungen ungenauer. Abbildung 3.1 zeigt, daß der zu Beginn der Integration gemachte Fehler noch für eine Weile den Gesamtfehler dominieren kann. Untersucht wurde hierfür die Fehlerfortpflanzung für das folgende System von gewöhnlichen Differentialgleichungen:

$$\begin{aligned} \dot{y}_1 &= -0.1\, y_1 \\ \dot{y}_2 &= y_1 - 10\, y_2 \\ \dot{y}_3 &= 10\, y_1 - 200\, y_3 \\ \dot{y}_4 &= -50\, y_2 - 5\, y_3 - 5\, y_4 \end{aligned} \tag{3.10}$$

mit Anfangswerten $y_i(0) = 1$, $i = 1, \ldots, 4$. Für das System (3.10) ist die analytische Lösung bekannt. Wir lösen das System mit Hilfe des BDF-Verfahrens DAESOL. Wir starten mit Ordnung 1 und Schrittweite 0.0002 bei einer angestrebten Genauigkeit von $TOL = 0.0001$. Für die Tests verwenden wir die im Integrator DAESOL adaptiv erzeugte Ordnungs- und Schrittweitenfolge. Um die Fehlerfortpflanzung nicht durch das nichtäquidistante Gitter zu verfälschen, wurde, nachdem das BDF-Verfahren die Ordnung 4 erreicht hatte, nach 3 weiteren Schritten die Schrittweite und Ordnung konstant gehalten.

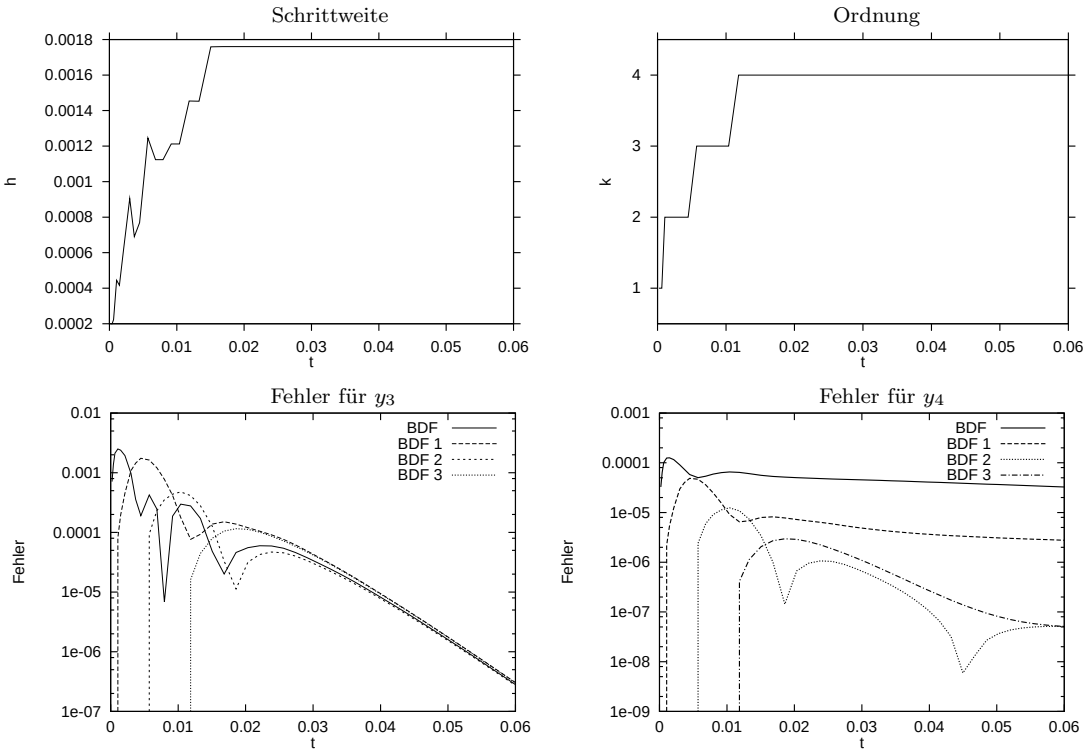


Abbildung 3.1: Schrittweiten- und Ordnungsfolge und Fehler für die Komponenten y_3 und y_4 für Beispiel (3.10).

Die Differenz zwischen der in DAESOL approximierten und der wahren Lösung

$$BDF = |y_{i,n+1} - y_i(t_{n+1})|$$

gibt den globalen Fehler von Komponente $i, i = 3, 4$, in jedem Schritt an. Um die Fortpflanzung des Fehlers untersuchen zu können, berechnen wir zudem die Trajektorien $y^j_{i,n+1}$, für die wir für die zurückliegenden Werte $y^j_{i,n}, \ldots, y^j_{i,n-k}$ anstelle der vom BDF-Verfahren approximierten Werte bis zur Ordnung $k = j$ die Werte der wahren Lösung $y_i(t_n), \ldots, y_i(t_{n-k})$ verwenden. Die Trajektorien $BDFj$, $j = 1, \ldots, 3$, in Abbildung 3.1 geben den zugehörigen Fehler an. Man sieht, daß der Fehler von BDF und $BDF\,1$ noch für eine Weile größer ist als der Fehler für die Approximationen y^j_i, $j = 2, 3$, bei denen die zurückliegenden Werte bis Ordnung $k = j$ durch exakte Werte ersetzt wurden.

Ein Neustart des BDF-Verfahrens ist meist nicht nur zu Beginn der Simulation nötig, sondern auch nach Unstetigkeiten, die zum Beispiel in einer physikalischen Änderung des Modells begründet liegen. Aber auch schon Unstetigkeiten aufgrund der Approximation von Steuerfunktionen durch stückweise definierte Polynome oder aufgrund der Mehrzieldiskretisierung im Optimierungskontext (siehe zum Beispiel Abschnitt 4.1.1) erfordern den Neustart des Verfahrens zu Beginn jedes Diskretisierungsintervalls. Ein Einschrittverfahren höherer Ordnung, das „zurückliegende“ Werte höherer Ordnung für das BDF-Verfahren bereitstellt, läßt somit eine große Einsparung an der Gesamtrechenzeit erwarten.

Die Idee, ein Runge-Kutta-Verfahren als Starter für ein Mehrschrittverfahren zu verwenden, findet sich zum ersten Mal bei Gear [Gea80]. Gear und – darauf aufbauend – Brankin et al. [BGS88] benutzten ein explizites Runge-Kutta-Verfahren zur Generierung der zurückliegenden Werte auch für die Integration von steifen Systemen. Ihre Begründung ist, daß auch steife Systeme zu Beginn der Integration meist eine transiente Phase haben, so daß die Schrittweite mehr durch die Nichtlinearität des Problems als durch die Stabilitätseigenschaften begrenzt wird. Allerdings trifft dies nicht auf alle steifen Systeme zu, vor allem im Zusammenhang mit Neustarts aufgrund von Unstetigkeiten zum Beispiel im Optimierungskontext. Wenn die Steifheit in der Startphase überwiegt, muß die Schrittweite – im Vergleich zum anschließenden impliziten Mehrschrittverfahren – für das explizite Runge-Kutta-Verfahren sehr klein gewählt werden. Zudem arbeitet das Mehrschrittverfahren sehr ineffizient, wenn die Schrittweite zu klein gewählt wurde, da zunächst viele Vergrößerungen der Schrittweite nötig sind und diese wiederum durch die Stabilitätsaussagen, wie in Abschnitt 2.1.3 beschrieben, begrenzt sind.

Von Schwerin und Bock [vSB95, vS97] haben ein explizites Runge-Kutta-Verfahren als Starter für ein Adams-Verfahren zur numerischen Behandlung von nicht-steifen Systemen in der Mechanik implementiert. Anders als bei Gear [Gea80] und Brankin et al. [BGS88] sind bei von Schwerin und Bock einige der internen Stufen von höherer Ordnung, so daß ein Schritt des Runge-Kutta-Verfahrens ausreicht, um alle zurückliegenden Werte bereitzustellen. Zudem ist die Fehlerschätzung des Runge-Kutta-Verfahrens auf diejenige des Adams-Verfahren zugeschnitten.

Im folgenden stellen wir ein implizites Runge-Kutta-Verfahren zur Generierung zurückliegender Werte höherer Ordnung für das in Kapitel 2 erläuterte BDF-Verfahren vor. Ähnlich wie bei von Schwerin und Bock [vSB95, vS97] sind einige der internen Stufen von höherer Ordnung, so daß mit einem Schritt des Einschrittverfahrens alle zurückliegenden Werte für einen Neustart des BDF-Verfahrens von der Ordnung 4 bereitgestellt werden können. Das Runge-Kutta-Verfahren wurde zudem so konstruiert, daß die internen Stufen innerhalb des Verfahrens, aber auch im Anschluß für das BDF-Verfahren, zur Lösung der nichtlinearen impliziten Gleichungssysteme dieselbe Iterationsmatrix verwenden. Außerdem ist die Fehlerschätzung – ähnlich wie bei von Schwerin und Bock – auf das BDF-Verfahren und die dortigen Fehlerformeln zugeschnitten.

3.2.1 Konstruktion des Runge-Kutta-Starters

Ein SDIRK-Verfahren (engl. *singly diagonally implicit Runge-Kutta method*) mit s internen Stufen zur Lösung des Anfangswertproblems (2.11) für linear implizite Systeme vom

Index 1 mit konsistenten Anfangswerten ist gegeben durch

$$\begin{aligned} A(t_n + c_i\,h, x_n + h\sum_{j=1}^{i-1} a_{ij}\,k_j + h\,\gamma\,k_i)\,k_i^y &= f(t_n + c_i\,h, x_n + h\sum_{j=1}^{i-1} a_{ij}\,k_j + h\,\gamma\,k_i) \\ 0 &= g(t_n + c_i\,h, x_n + h\sum_{j=1}^{i-1} a_{ij}\,k_j + h\,\gamma\,k_i) \\ x_{n+1} &= x_n + h\sum_{i=1}^{s} b_i\,k_i \\ \text{mit } k_i &= \left((k_i^y)^T, (k_i^z)^T\right)^T. \end{aligned} \tag{3.11}$$

Allgemein läßt sich das Verfahren schreiben als ein sogenanntes *Butcher-Tableau*, siehe Tabelle 3.1.

γ	γ			
c_2	a_{21}	γ		
$\vdots$	$\vdots$		$\ddots$	
c_s	a_{s1}	$\cdots$	$a_{s,s-1}$	γ
	b_1	$\cdots$	b_{s-1}	b_s
	$\hat{b}_1$	$\cdots$	$\hat{b}_{s-1}$	$\hat{b}_s$

Tabelle 3.1: Butcher-Tableau für ein SDIRK-Verfahren mit s internen Stufen

Für die Fehlerschätzung verwenden wir ein eingebettetes Verfahren. Die Idee ist dabei, neben der Approximation x_{n+1} eine weitere Approximation $\hat{x}_{n+1}$ zu berechnen; die Differenz ergibt eine Schätzung des Fehlers für das weniger genaue Ergebnis. Normalerweise wird die zusätzliche Approximation über unterschiedliche Werte der b_i erzeugt. In der Regel ist die Ordnung der Approximation um eins höher oder niedriger als das ursprüngliche Verfahren.

Wendet man das Verfahren (3.11) auf das Anfangswertproblem

$$\dot{x} = \lambda\,x, \quad x(t_0) = x_0,$$

an, so erhält man $x_1 = R(h\,\lambda)\,x_0$ mit

$$R(z) = 1 + z\,b^T\,(I - z\,\mathcal{A})^{-1}\,e$$

und $b^T = (b_1, \ldots, b_s)$, $\mathcal{A} = (a_{ij})_{i,j=1}^s$, $e = (1, \ldots, 1)^T$.

Definition 3.2.1 (Stabilitätsfunktion)
Die Funktion $R(z)$ heißt Stabilitätsfunktion *des impliziten Runge-Kutta-Verfahrens (3.11).*

Konsistenz- und Konvergenzbedingungen

Konsistenzbedingungen für das Runge-Kutta-Verfahren erhält man, indem man die Taylorreihenentwicklung der exakten Lösung mit derjenigen der approximierten Lösung vergleicht. Für ein Runge-Kutta-Verfahren der Ordnung 4 mit s internen Stufen ergeben sich daraus folgende Bedingungen[1]:

$$\text{Ordnung } 1: \quad \sum_{i=1}^{s} b_i = 1 \tag{3.12a}$$

$$\text{Ordnung } 2: \quad \sum_{i=1}^{s} b_i\, c_i = 1/2 \tag{3.12b}$$

$$\text{Ordnung } 3: \quad \sum_{i=1}^{s} b_i\, c_i^2 = 1/3 \tag{3.12c}$$

$$\sum_{i,j=1}^{s} b_i\, a_{ij}\, c_j = 1/6 \tag{3.12d}$$

$$\text{Ordnung } 4: \quad \sum_{i=1}^{s} b_i\, c_i^3 = 1/4 \tag{3.12e}$$

$$\sum_{i,j=1}^{s} b_i\, c_i\, a_{ij}\, c_j = 1/8 \tag{3.12f}$$

$$\sum_{i,j=1}^{s} b_i\, a_{ij}\, c_j^2 = 1/12 \tag{3.12g}$$

$$\sum_{i,j,k=1}^{s} b_i\, a_{ij}\, a_{jk}\, c_k = 1/24 \tag{3.12h}$$

Dabei wird vorausgesetzt, daß die internen Stufen konsistent sind, das heißt

$$\text{Ordnung } 1: \sum_{j=1}^{s} a_{ij} = c_i, \qquad i = 1, \ldots, s. \tag{3.13a}$$

Zudem sollen die internen Stufen $4-7$ des Verfahrens von der Ordnung 3 sein, wofür die zusätzlichen Bedingungen erfüllt sein müssen:

$$\text{Ordnung } 2: \sum_{j=1}^{s} a_{ij}\, c_j = \frac{c_i^2}{2}, \qquad i = 4, \ldots, 7, \tag{3.13b}$$

$$\text{Ordnung } 3: \sum_{j=1}^{s} a_{ij}\, c_j^2 = \frac{c_i^3}{3}, \qquad i = 4, \ldots, 7, \tag{3.13c}$$

[1] Im Falle von SDIRK-Verfahren ist $a_{ii} = \gamma$, $i = 1, \ldots, s$, und $a_{ij} = 0$ für $j > i$.

$$\sum_{j,k=1}^{s} a_{ij}\, a_{jk}\, c_k = \frac{c_i^3}{6}, \quad i = 4, \ldots, 7. \tag{3.13d}$$

Satz 3.2.1 (Konvergenz von Runge-Kutta-Verfahren)
Sei das Anfangswertproblem (2.11) für eine DAE vom Index 1 gegeben und die Anfangswerte seien konsistent. Wendet man auf das Anfangswertproblem ein Runge-Kutta-Verfahren der Form (3.11) von der Ordnung p mit internen Stufen der Ordnung q und regulärer Matrix $\mathcal{A}$ an, so gilt für den globalen Fehler der numerischen Lösung

$$y_n - y(t_n) = \mathcal{O}(h^p), \quad z_n - z(t_n) = \mathcal{O}(h^r), \quad t_n - t_0 = n \cdot h = const$$

mit

a) $r = p$ *für steif genaue Verfahren, das heißt* $a_{sj} = b_j$, $j = 1, \ldots, s$,

b) $r = \min(p, q+1)$, *falls* $-1 \leq R(\infty) < 1$,

c) $r = \min(p-1, q)$, *falls* $R(\infty) = 1$.

d) Falls $|R(\infty)| > 1$ *ist, so divergiert die numerische Lösung.*

Beweis: Siehe zum Beispiel Hairer und Wanner [HW96b].

Konstruktion des Verfahrens

Nach Satz 3.2.1 gibt es zwei grundsätzliche Möglichkeiten zur Konstruktion eines SDIRK-Verfahrens der Ordnung 4 mit mindestens 4 internen Stufen der Ordnung 3:

a) Implementiere ein steif genaues Verfahren, das heißt

$$a_{sj} = b_j,\ j = 1, \ldots, s.$$

b) Oder implementiere ein Verfahren, für das für die Stabilitätsfunktion $-1 \leq R(\infty) < 1$ gilt und bei dem nur diejenigen b_i ungleich Null sind, deren zugehörige interne Stufen i von der Ordnung $p-1$, hier also 3, sind.

Geht man von insgesamt 7 internen Stufen aus, so führt die erste Möglichkeit zusammen mit den Ordnungsbedingungen allerdings auf ein überbestimmtes System. Im Fall *b*) können alle Bedingungen mit 7 internen Stufen erfüllt werden.

Lösen des nichtlinearen Gleichungssystems

Wendet man das SDIRK-Verfahren auf ein Anfangswertproblem für DAEs vom Index 1 der Form (2.11) an, so erfordert dies in jeder internen Stufe das Lösen des nichtlinearen Gleichungssystems

$$\begin{aligned} F(k_i) &= \begin{pmatrix} F_1(k_i) \\ F_2(k_i) \end{pmatrix} \\ &= \begin{pmatrix} A(t_n + c_i\,h, x_n + h\,\sum_{j=1}^{i} a_{ij}\,k_j)\,k_i^y - f(t_n + c_i\,h, x_n + h\,\sum_{j=1}^{i} a_{ij}\,k_j) \\ g(t_n + c_i\,h, x_n + h\,\sum_{j=1}^{i} a_{ij}\,k_j) \end{pmatrix} \end{aligned} \tag{3.14}$$

Die Berechnung erfolgt wiederum mit einem vereinfachten Newton-Verfahren:

$$k_i^{(m+1)} = k_i^{(m)} + \Delta k_i^{(m)}.$$

Die Iterierte $\Delta k_i^{(m)}$ löst das lineare Gleichungssystem

$$J_{RK} \cdot \Delta k_i^{(m)} = F(k_i^{(m)})$$

mit genäherter Iterationsmatrix

$$J_{RK} \approx \left.\frac{\partial F}{\partial k_i}\right|_{k_i^{(m)}} = \left.\begin{pmatrix} A + h\,\gamma\,A_y\,k_i^y - h\,\gamma\,f_y & h\,\gamma\,A_z\,k_i^y - h\,\gamma\,f_z \\ h\,\gamma\,g_y & h\,\gamma\,g_z \end{pmatrix}\right|_{k_i^{(m)}}.$$

Durch die Wahl des SDIRK-Verfahrens wurde schon sichergestellt, daß innerhalb eines Runge-Kutta-Schrittes in jeder Stufe für das vereinfachte Newton-Verfahren die gleiche Iterationsmatrix verwendet werden kann. Jedoch soll die Berechnung und Zerlegung von J_{RK} auch im Anschluß für das BDF-Verfahren wiederverwendet werden können. Deshalb skalieren wir die Zeilen von F um

$$\hat{F}(k_i) = \begin{pmatrix} \hat{F}_1(k_i) \\ \hat{F}_2(k_i) \end{pmatrix} = \begin{pmatrix} -\frac{h_{BDF}}{h_{RK}\cdot\gamma} F_1(k_i) \\ -\frac{1}{h_{RK}\cdot\gamma} F_2(k_i) \end{pmatrix}.$$

und lösen das nichtlineare Gleichungssystem in $\hat{F}$.

Um die Konvergenzaussagen für das BDF-Verfahren zu erfüllen, sollen die internen Stufen der Ordnung 3, die als zurückliegende Werte für das BDF-Verfahren dienen, möglichst äquidistant über den Gesamtschritt des Runge-Kutta-Verfahrens verteilt werden, also

$$c_{i+3} = \frac{i}{5} \cdot h_{RK},\ i = 1, \ldots, 4. \tag{3.15}$$

Dann ist $h_{BDF} = \frac{1}{5} \cdot h_{RK}$ und für den führenden Koeffizienten im BDF-Verfahren gilt dann

$$\alpha_0 = -\frac{25}{12}.$$

Mit

$$\gamma = -\frac{h_{BDF}}{h_{RK}\,\alpha_0} = \frac{12}{125} \tag{3.16}$$

ist die zugehörige Iterationsmatrix $\hat{J}_{RK}$ gerade von der gleichen Form wie diejenige für das vereinfachte Newton-Verfahren im BDF-Verfahren (siehe (2.14)). Die Iterationsmatrix kann für das vereinfachte Newton-Verfahren im BDF-Verfahren wiederverwendet werden.

Insgesamt erhalten wir mit den 8 Ordnungsbedingungen (3.12) für die äußere Stufe und den 19 Bedingungen (3.13) für die internen Stufen, den 4 Bedingungen (3.15) für die Stützstellen c_i, der Bedingung (3.16) für γ, den Bedingungen an die b_i, deren interne Stufe i nicht von der Ordnung 3 ist

$$b_i = 0, \quad i = 1, \ldots, 3,$$

und der Bedingung an die Stabilitätsfunktion

$$R(\infty) = 0$$

36 Bedingungen für 36 Unbekannte.

Sukzessives Auflösen der Bedingungen mit Maple [GH93, Kof96] ergibt das in Tabelle 3.2 aufgeführte Tableau für das implementierte Runge-Kutta-Verfahren.

	.096	.096						
	.2	.104	.096					
	.3	.131045	.072955	.096				
$x_1 \leftarrow$	.2	.219960	$-.144718$	.028758	.096			
$x_2 \leftarrow$	.4	.160885	$-.051240$	$-.024680$	.219035	.096		
$x_3 \leftarrow$	.6	.176169	$-.237690$	.124977	.303425	.137120	.096	
$x_4 \leftarrow$	.8	.182252	$-.346544$	.213542	.354749	.1	.2	.096
	x_{n+1}	0	0	0	.45833	.04167	.04167	.45833
	$\hat{x}_{n+1}$	$\hat{b}_1$	$\hat{b}_2$	$\hat{b}_3$	$\hat{b}_4$	$\hat{b}_5$	$\hat{b}_6$	$\hat{b}_7$

Tabelle 3.2: Werte der Koeffizienten für das SDIRK-Verfahren zur Generierung von zurückliegenden Werten für das BDF-Verfahren.

Die Variablen x_i, $i = 1, \ldots, 4$, geben dabei die Werte höherer Ordnung an den internen Stufen an, die im Anschluß als zurückliegende Werte für das BDF-Verfahren verwendet werden.

3.2.2 Fehlerschätzung und Schrittweitensteuerung

Für die Fehlerschätzung wählen wir ein eingebettetes Verfahren der Ordnung 3. Dieses soll genau auf die Fehlerformel (2.21) im BDF-Verfahren zugeschnitten sein.

Startet man das BDF-Verfahren mit Ordnung 4 im Anschluß an das SDIRK-Verfahren, so verwenden wir die Werte an den internen Stufen der Ordnung 3 als zurückliegende Werte:

$$x_i = x_0 + h_{RK}\,(a_{j1}\,k_1 + \cdots + a_{j,j-1}\,k_{j-1} + \gamma\,k_j), \quad i = 1, \ldots, 4,\ j := i + 3.$$

Da die Schrittweite konstant ist, vereinfachen sich die Terme in der Fehlerformel (2.21)

$$\begin{aligned}\psi_4(5) &= 4! \cdot h_{BDF}^4 \\ \nabla^5 x_5 &= \frac{1}{5! \cdot h_{BDF}^5} (x_5 - 5\,x_4 + 10\,x_3 - 10\,x_2 + 5\,x_1 - x_0).\end{aligned}$$

Für $k = 4$ und $n = 4$ ergibt die Fehlerschätzung für den Schritt $n + 1$:

$$\begin{aligned}E_4(5) &= h_{BDF} \cdot \psi_4(5) \cdot \|\nabla^5 x_5\| \\ &= \frac{1}{5} \|x_5 - 5\,x_4 + 10\,x_3 - 10\,x_2 + 5\,x_1 - x_0\|.\end{aligned}$$

Mit

$$\begin{aligned}\hat{x}_5 &= x_0 - 5\,x_1 + 10\,x_2 - 10\,x_3 + 5\,x_4 \\ &=: \hat{b}_1\,k_1 + \cdots + \hat{b}_7\,k_7\end{aligned}$$

erhalten wir die Fehlerkoeffizienten $\hat{b}$ für das eingebettete Verfahren:

$$\hat{b}^T = \begin{pmatrix}\text{-.341381} & .855374 & \text{-.572645} & .449848 & .088804 & .04 & 0.48\end{pmatrix}$$

Schrittweitensteuerung

Die Schätzung einer Ordnung und Schrittweite für den nächsten Schritt im BDF-Verfahren erfolgt analog zu der in Abschnitt 2.3.3 beschriebenen.

Für die Berechnung einer neuen Schrittweite, falls das SDIRK-Verfahren nicht konvergiert hat, verwenden wir analog zum BDF-Verfahren die aktuelle Fehlerschätzung zur Reduzierung der Schrittweite, siehe Formel (2.26).

Kapitel 4

Optimierungsprobleme bei der Parameterschätzung

Die Modellierung von Prozessen aus Chemie und Verfahrenstechnik führt in vielen Fällen auf ein System von differentiell-algebraischen Gleichungen oder partiellen Gleichungen teilweise gekoppelt mit algebraischen Gleichungen. Die partiellen Gleichungen werden oft mit Hilfe der Linienmethode oder einem Finite-Element-Ansatz im Ort diskretisiert und in ein DAE-System überführt.

Typisch für viele Modelle aus Chemie und Verfahrenstechnik ist, daß sie unbekannte Parameter enthalten, die nicht direkt gemessen werden können. So wird zum Beispiel bei der Modellierung von chemischen Reaktionsgleichungen sehr oft ein Arrhenius-Ansatz $k = f\, e^{-\frac{E_a}{RT}}$ als Approximation der Reaktionsgeschwindigkeit verwendet. Der Frequenzfaktor f und die Aktivierungsenergie E_a sind die von der Reaktion abhängigen Parameter, R stellt die Gaskonstante dar und T die Temperatur im Reaktor. Werden für das chemische Reaktionssystem alle Elementarreaktionen mitmodelliert, so sind die hierfür benötigten Parameter in der Regel in Tabellen vohanden. Der Preis hierfür ist allerdings, daß zunächst ein kompliziertes Reaktionssystem aufgestellt werden muß und das resultierende DAE-System zudem sehr groß sein kann.

Der weiter verbreitete Ansatz ist, nur die Hauptreaktionen in Betracht zu ziehen und ein vereinfachtes Modell zu verwenden. Hierfür erhält man in der Regel Differentialgleichungen für nur wenige Spezies. Die Reaktionsgeschwindigkeit wird in der Regel wiederum mit dem Arrhenius-Ansatz modelliert, eventuell werden noch die Reaktionsordnungen als zusätzliche Parameter eingeführt. Für diese Parameter liegen in der Regel allerdings nur noch grobe und oft unzureichende Schätzungen vor.

Da die Parameter selbst normalerweise nicht gemessen werden können, versucht man, aus vorhandenen Meßdaten des chemischen Prozesses Rückschlüsse auf die Parameter zu ziehen. Das Ziel ist, die Parameter so zu schätzen, daß das zugehörige Modell die Meßdaten „möglichst gut“ approximiert.

Die Meßdaten η_{ij} können Messungen der Zustandsvariablen selbst sein (etwa die Temperatur oder die Konzentration einer Spezies) oder aber allgemeine Funktionen b_i auf den

Zustandsvariablen, den Parametern oder den Steuergrößen des Prozesses

$$\eta_{ij} \;=\; b_i(t_j, x(t_j), p, q) + \varepsilon_{ij}$$

wie etwa der pH-Wert oder der Druck, wenn er nicht als Zustandsvariable mitmodelliert wurde.

Wir betrachten im folgenden Parameterschätzprobleme, bei denen die Meßfehler ε_{ij} unabhängig und normalverteilt sind mit Erwartungswert 0 und Varianz σ_{ij}^2

$$\varepsilon_{ij} \sim \mathcal{N}(0, \sigma_{ij}^2).$$

Für den Fall nicht-normalverteilter Meßfehler und falls die Meßdaten starke Ausreißer aufweisen, wird zum Beispiel von Huber [Hub96] eine robuste Parameterschätzung im Sinne eines l_1-Funktionals oder eines l_p-Funktionals mit $1 < p < 2$ vorgeschlagen.

Für das Parameterschätzproblem nehmen wir weiterhin an, daß die unabhängige Variable (der Zeitpunkt der Messung beziehungsweise der Ort) bekannt ist. Schlöder [Sch88] und Stortelder [Sto97] untersuchten Parameterschätzprobleme bei ODE- bzw. DAE-Systemen für den Fall, daß die Meßfehler der unabhängigen Variable t nicht zu vernachlässigen sind. Das Least-Squares-Problem führt dann auf eine orthogonale Abstandsminimierung.

Zur Schätzung der Parameter minimieren wir die Summe der quadratischen Abweichungen zwischen den Messungen η_{ij} und der Modellantwort b_i, gewichtet mit der Standardabweichung σ_{ij}

$$\min_{p,x} \sum_{i,j} w_{ij} \, \frac{(\eta_{ij} - b_i(t_j, x(t_j), p, q))^2}{\sigma_{ij}^2}. \tag{4.1}$$

Die Gewichte $w_{ij} \in \{0, 1\}$ beschreiben, ob eine Messung durchgeführt wurde oder nicht. Kann eine Messung zu einem Zeitpunkt mehrmals durchgeführt werden, so kann auch $w_{ij} \in \{0, N_{w_{ij}}\}$, $N_{w_{ij}} \in \mathbb{N}$, gelten. Die Zustandsvariablen $x(t) = (y(t), z(t)) \in \mathbb{R}^{n_y} \times \mathbb{R}^{n_z}$, $n_y, n_z \geq 0$, $n_x = n_y + n_z$, erfüllen dabei die Lösung des zugrundeliegenden DAE-Systems. Wir konzentrieren uns im folgenden auf Parameterschätzprobleme, deren Dynamik durch ein DAE-System vom Index 1 der Form

$$\begin{aligned} A(t, y, z, p, q, u)\, \dot{y} \;&= f(t, y, z, p, q, u) \\ 0 \;&= g(t, y, z, p, q, u) \end{aligned} \tag{4.2}$$

modelliert wird, wobei $t \in \mathbb{R}$ die Zeit beziehungsweise den Ort beschreibt, $p \in \mathbb{R}^{n_p}$ den Vektor der unbekannten Parameter und $q \in \mathbb{R}^{n_q}$ und $u : [t_0, t_f] \to \mathbb{R}^{n_u}$ konstante Steuergrößen beziehungsweise zeitlich oder vom Ort abhängige Steuerfunktionen, die den Einstellungen bei der Fahrweise des Prozesses entsprechen. Steuergrößen sind zum Beispiel Anfangskonzentrationen oder die Anfangstemperatur des Prozesses im Reaktor, Steuerfunktionen zum Beispiel ein Zulaufprofil oder die Steuerung der Temperatur. Für die Parameterschätzung sind die Inputgrößen q und $u(t)$ für die Versuchsdurchführung sowie die Gewichte fest. Sie werden aber in Kapitel 5 für die Aufstellung des Versuchsplanungsproblems benötigt und sind deshalb schon hier eingeführt.

Eventuell können weitere Gleichungsnebenbedingungen wie etwa Anfangsbedingungen, Randbedingungen oder Innere-Punkte-Bedingungen auftreten

$$d(x(t_0), \ldots, x(t_f), p, q) = 0\,. \tag{4.3}$$

Einen Überblick zur Parameterschätzung bei nichtlinearen Regressionsmodellen sowie der statistischen Analyse der geschätzten Parameter geben zum Beispiel die Bücher von Bates und Watts [BW88] und von Seber und Wild [SW89]. Björck [Bjö96] beschreibt numerische Methoden zur Lösung von linearen und nichtlinearen Ausgleichsproblemen. Nichtlineare Ausgleichsprobleme werden meist mit Gauß-Newton-Verfahren gelöst, die oft schon in der Einführungsliteratur zur Numerik behandelt werden (siehe zum Beispiel die Bücher von Stoer und Bulirsch [SB73] und von Dennis und Schnabel [DS83]). Einen Überblick von Programmpaketen zur Lösung von Optimierungsproblemen und insbesondere auch von nichtlinearen Least-Squares-Problemen geben Moré und Wright [MW93].

Bard [Bar74] beschreibt neben der Lösung von Parameterschätzproblemen für lineare und nichtlineare Regressionsmodelle und deren statistische Analyse auch Verfahren zur numerischen Lösung von Parameterschätzproblemen bei nichtlinearen dynamischen Systemen. Programmpakete hierzu basieren sehr oft auf dem Single-Shooting-Ansatz (siehe zum Beispiel den Code von Schittkowski [Sch99a, Sch99b]), jedoch besitzt der im folgenden beschriebene Multiple-Shooting-Ansatz im Gegensatz dazu Vorteile im Hinblick auf Stabilität und auf die Verwendung von Vorinformationen zum Beispiel aus den Meßdaten zur Verbesserung der Konvergenz.

Im folgenden beschreiben wir Methoden zur Lösung des Parameterschätzproblems (4.1, 4.3) mit einem DAE-System (4.2) als Nebenbedingung, die auf dem Mehrzielverfahren beruhen (siehe hierzu die Methoden von Bock und Schlöder [Boc81, BS86, Boc87, Sch88] zur Lösung von Parameterschätzproblemen bei ODE-Systemen). Wir parametrisieren die unendlich-dimensionale Lösung des DAE-Systems. Das resultierende nichtlineare endlich-dimensionale Ausgleichsproblems wird mit dem verallgemeinerten Gauß-Newton-Verfahren PARFIT von Bock [Boc81, Boc87] gelöst. Für die effiziente Lösung des Parameterschätzproblems kann in vielen Fällen durch geschicktes Ausnutzen von Gleichungsnebenbedingungen ein reduzierter Ansatz verwendet werden, wie er zum ersten Mal von Schlöder [Sch88] für Parameterschätzprobleme bei ODE-Systemen eingeführt wurde. Soll das Parameterschätzproblem die Meßdaten aus mehreren Experimenten berücksichtigen, können die dabei speziell auftretenden Strukturen zusätzlich ausgenutzt werden (siehe Schlöder und Bock [SB83, Sch88] und von Schwerin [vS98], wiederum für den ODE-Fall).

Zuletzt gehen wir auf die statistische Analyse der geschätzten Parameter ein und geben eine Berechnungsvorschrift für eine näherungsweise Kovarianzmatrix an. Darauf aufbauend werden im nächsten Kapitel zur Gewinnung der für die Parameterschätzung notwendigen Meßdaten Methoden der optimalen Versuchsplanung bei DAE-Systemen vorgestellt. Da die Versuchsplanung in der Regel aus einer sukzessiven Vorgehensweise aus Planung von neuen Experimenten, Erhebung der Meßdaten, Parameterschätzung und Planung weiterer Experimente besteht, basieren die der Versuchsplanung zugrundeliegenden Parameterschätzprobleme meist auf Meßdaten aus mehreren Experimenten. Die Strukturen

aufgrund des Mehrfachexperiment-Ansatzes werden sowohl für die Parameterschätzung als auch anschließend für die Versuchsplanung ausgenutzt. Um die Dimension der Kovarianzmatrix (und somit den Rechenaufwand) gering zu halten, baut die Versuchsplanung zudem in der Regel auf dem in Abschnitt 4.4 beschriebenen reduzierten Ansatz für das zugrundeliegende Parameterschätzproblem auf.

4.1 Numerische Lösung des Parameterschätzproblems

4.1.1 Mehrzielmethode

Basierend auf den Methoden des Mehrfachschießverfahrens für Zweipunkt-Randwertprobleme wie sie von Bulirsch [Bul71], Stoer und Bulirsch [SB73] und Deuflhard [Deu74] eingeführt und behandelt wurden, wird die unendlich-dimensionale Lösung $x(t)$ des DAE-Systems parametrisiert und in eine Funktion mit endlich vielen Freiheitsgraden überführt. Von Bock [Boc78] wurde die Mehrzielmethode erstmals auf die Klasse von Mehrpunkt-Randwertproblemen angewendet.

Wir diskretisieren den Beobachtungszeitraum $[t_0, t_f]$ in

$$t_0 = \tau_0 < \ldots < \tau_m = t_f \,.$$

Auf jedem Teilintervall $[\tau_j, \tau_{j+1}]$, $j = 0, \ldots, m-1$, führen wir zusätzliche Variablen s_j^y, s_j^z ein und lösen jeweils ein Anfangswertproblem für eine DAE in relaxierter Form

$$\begin{aligned} A(t, y, z, p, q, u)\, \dot{y} &= f(t, y, z, p, q, u) && \text{(4.4a)} \\ 0 &= g(t, y, z, p, q, u) - \vartheta(t)\, g(\tau_j, y(\tau_j), z(\tau_j), p, q, u) && \text{(4.4b)} \end{aligned}$$

mit Anfangswerten $y(\tau_j) = s_j^y$ und $z(\tau_j) = s_j^z$. Die relaxierte Formulierung der algebraischen Gleichungen (4.4b) des DAE-Systems garantiert konsistente Anfangswerte auf jedem Teilintervall. Von Bock et al. [BES88] wurde die relaxierte Formulierung im Optimierungskontext für $\vartheta(t) \equiv 1$ für DAEs vom Index 1 eingeführt. Eine Erweiterung für DAEs von höherem Index und einer allgemeineren Wahl von $\vartheta(t)$ findet sich in Schulz et al. [SBS98], siehe auch Abschnitt 3.1.2. Die Konsistenz der ursprünglichen algebraischen Gleichungen wird über das Optimierungsverfahren erzwungen und ist im Lösungspunkt erfüllt.

Wir erhalten ein endlich-dimensionales Optimierungsproblem in den Variablen $s = (s_0^y, s_0^z, \ldots, s_m^y, s_m^z)$ und p

$$\begin{aligned} \min_{s,p} \quad & \sum_{i,j} w_{ij}\, \frac{(\eta_{ij} - b_i(t_j, x(t_j; s, p), p))^2}{\sigma_{ij}^2} \\ & =: \|r(s,p)\|_2^2 \end{aligned} \qquad (4.5)$$

$$\text{so daß} \qquad \bar{d}(s,p) = 0\,. \tag{4.6}$$

Stetigkeitsbedingungen für die differentiellen Variablen

$$h_i(s_i, s_{i+1}, p) := y(\tau_{i+1}; \tau_i, s_i, p) - s_{i+1}^y = 0,\ i = 0, \ldots, m-1, \tag{4.7}$$

zusammen mit Konsistenzbedingungen zur Bestimmung der algebraischen Variablen

$$g_i(s_i, p) := g(\tau_i, s_i, p, q, u) = 0,\ i = 0, \ldots, m, \tag{4.8}$$

garantieren, daß die Trajektorie im Lösungspunkt des Optimierungsproblems eine stetige Lösung der ursprünglichen DAE (4.2) auf dem gesamten Intervall $[t_0, t_f]$ ist.

Bemerkung 4.1.1 (Vorteile der Mehrzielmethode)
Der Vorteil der Mehrzieldiskretisierung liegt in der freien Wahl des Mehrzielgitters und der Startschätzungen s_j. Dies erlaubt dem Benutzer, Vorinformationen – z. B. aus den Meßdaten – bei den Startschätzungen an den Gitterpunkten zu berücksichtigen. Der Einfluß von schlechten Schätzungen der Parameter wird dadurch gemildert. Darüberhinaus ist der Mehrzielansatz numerisch stabil (siehe Bock [Boc87]).

4.1.2 Verallgemeinertes Gauß-Newton-Verfahren

Das resultierende Least-Squares-Problem (4.5) zusammen mit den Nebenbedingungen (4.6), (4.7) und (4.8) lösen wir mit einem verallgemeinerten Gauß-Newton-Verfahren:

Beginne mit einem Startwert $\nu^{(0)} = (s^{(0)}, p^{(0)})$. Das Gauß-Newton-Verfahren basiert auf einer Sequenz von linearen Approximationen. Sei $\nu^{(k)} = (s^{(k)}, p^{(k)})$ die aktuelle Approximation, dann berechne eine Korrektur $\Delta\nu^{(k)}$ als Lösung des linearisierten Ausgleichsproblems

$$\begin{aligned} &\min_{\Delta\nu} \|r(\nu) + R\,\Delta\nu\|_2^2 \\ \text{so daß } &\bar{d}(\nu) + D\,\Delta\nu = 0 \\ &h(\nu) + H\,\Delta\nu = 0 \\ &g(\nu) + G\,\Delta\nu = 0\,, \end{aligned} \tag{4.9}$$

wobei $h = (h_0^T, \ldots, h_{m-1}^T)^T$ und $g = (g_0^T, \ldots, g_m^T)^T$.

Die Jacobi-Matrix J ist dann von der Form

$$J = \begin{pmatrix} J_1 \\ \cdots \\ \\ \\ \\ J_2 \\ \\ \\ \\ \end{pmatrix} = \begin{pmatrix} R \\ \cdots \\ D \\ \cdots \\ \\ H \\ \\ \cdots \\ \\ G \\ \\ \end{pmatrix} = \begin{pmatrix} R^{s_0} & \cdots & & R^{s_m} & R^p \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ D^{s_0} & \cdots & & D^{s_m} & D^p \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ H_0^{s_0} & H_0^{s_1} & & & H_0^p \\ & \ddots & \ddots & & \vdots \\ & & H_{m-1}^{s_{m-1}} & H_{m-1}^{s_m} & H_{m-1}^p \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ G_0^{s_0} & & & & G_0^p \\ & \ddots & & & \vdots \\ & & & G_m^{s_m} & G_m^p \end{pmatrix} \tag{4.10}$$

mit

$$
\begin{aligned}
R^{s_i} &= \frac{\partial r}{\partial s_i}, \; i=0,\ldots,m, & R^{p} &= \frac{\partial r}{\partial p}, \\
D^{s_i} &= \frac{\partial \bar{d}}{\partial s_i}, \; i=0,\ldots,m, & D^{p} &= \frac{\partial \bar{d}}{\partial p}, \\
H_i^{s_i} &= \frac{\partial h_i}{\partial s_i}, \; i=0,\ldots,m-1, & H_i^{p} &= \frac{\partial h_i}{\partial p}, \; i=0,\ldots,m-1, \\
H_i^{s_{i+1}} &= (-I_{n_y}, 0_{n_y \times n_z}), \; i=0,\ldots,m-1, & & \\
G_i^{s_i} &= \frac{\partial g_i}{\partial s_i}, \; i=0,\ldots,m, & G_i^{p} &= \frac{\partial g_i}{\partial p}, \; i=0,\ldots,m-1.
\end{aligned}
$$

4.1.3 Kondensierung

Das Ausgleichsproblem (4.9) ist zunächst sehr groß, weist aber ganz spezielle Strukturen auf. Deshalb wird nicht das gesamte Problem direkt gelöst, sondern nacheinander die Strukturen ausgenutzt, was letztendlich auf ein Ausgleichsproblem von weitaus geringerer Dimension führt (siehe zum Beispiel den Kondensieralgorithmus von Bock [Boc83] für Parameterschätzprobleme bei ODE-Systemen oder eine ähnliche Vorgehensweise von Leineweber [Lei99] für Optimierungsprobleme bei DAE-Systemen).

Wir eliminieren zunächst die zu Δs_i^z gehörigen Blöcke

$$
\Delta s_i^z = -\left(G_i^{s_i^z}\right)^{-1} \left(G_i^{s_i^y} \, \Delta s_i^y + G_i^p \, \Delta p + g_i\right). \tag{4.11}
$$

Dies ist möglich, da aufgrund der Index-1-Bedingung die Matrix $G_i^{s_i^z}$ regulär ist. Formal können wir die Elimination auffassen als Multiplikation der Jacobi-Matrix J mit der Matrix

$$
P = \begin{pmatrix}
\tilde{G}_0 & & & \tilde{G}_0^p \\
& \ddots & & \vdots \\
& & \tilde{G}_m & \tilde{G}_m^p \\
& & & I_{n_p}
\end{pmatrix}
$$

mit

$$
\tilde{G}_i = \begin{pmatrix} I_{n_y} \\ -(G_i^{s_i^z})^{-1} \, G_i^{s_i^y} \end{pmatrix} \;\text{ und }\; \tilde{G}_i^p = \begin{pmatrix} 0_{n_y \times n_p} \\ -(G_i^{s_i^z})^{-1} \, G_i^p \end{pmatrix}, \quad i = 0, \ldots, m.
$$

Wir erhalten ein reduziertes Ausgleichsproblem in den Variablen $((\Delta s_0^y)^T, (\Delta s_1^y)^T, \ldots, (\Delta s_m^y)^T, (\Delta p)^T)^T$ mit Jacobi-Matrix $\tilde{J}$ und neuer rechter Seite rhs:

$$
\tilde{J} = \begin{pmatrix}
\tilde{R}_0 & \cdots & \tilde{R}_m & \tilde{R}^p \\
\tilde{D}_0 & \cdots & \tilde{D}_m & \tilde{D}^p \\
\tilde{H}_0 & -I_{n_y} & & \tilde{H}_0^p \\
& \ddots & \ddots & \vdots \\
& & \tilde{H}_{m-1} & -I_{n_y} \quad \tilde{H}_{m-1}^p
\end{pmatrix}, \qquad rhs = -\begin{pmatrix} \tilde{r} \\ \tilde{d} \\ \tilde{h}_0 \\ \vdots \\ \tilde{h}_{m-1} \end{pmatrix}
$$

mit

$$\tilde{R}_i = R_i^{s_i^y} - R_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^{s_i^y},\ i=0,\ldots,m, \qquad \tilde{R}^p = R^p - \sum_{i=0}^{m} R_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^p,$$

$$\tilde{D}_i = D_i^{s_i^y} - D_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^{s_i^y},\ i=0,\ldots,m, \qquad \tilde{D}^p = D^p - \sum_{i=0}^{m} D_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^p,$$

$$\tilde{H}_i = H_i^{s_i^y} - H_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^{s_i^y},\ i=0,\ldots,m-1, \qquad \tilde{H}_i^p = H_i^p - H_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,G_i^p,\ i=0,\ldots,m-1,$$

$$\tilde{r} = r - \sum_{i=0}^{m} R_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,g_i,$$

$$\tilde{d} = \bar{d} - \sum_{i=0}^{m} D_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,g_i,$$

$$\tilde{h}_i = h_i - H_i^{s_i^z}\,(G_i^{s_i^z})^{-1}\,g_i,\ i=0,\ldots,m-1.$$

Das reduzierte System kann zum Beispiel wiederum mit Hilfe einer Block-Gauß-Elimination (siehe zum Beispiel Bock [Boc87] oder Schlöder [Sch88]) gelöst werden. Dabei werden zunächst die zu den Variablen Δs_i^y, $i = m, \ldots, 1$, gehörigen Blöcke sukzessive eliminiert:

Sei $u_1^m := \tilde{r}$, $u_2^m := \tilde{d}$, $E_1^m := \tilde{R}_m$, $E_2^m := \tilde{D}_m$, $E_1^{p,m} := \tilde{R}^p$, $E_2^{p,m} := \tilde{D}^p$.

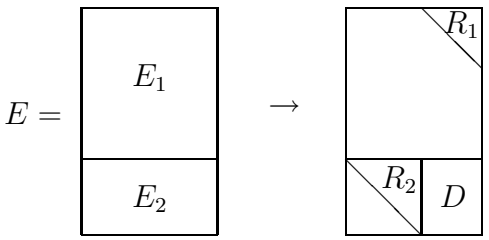
Berechne für $j = m, \ldots, 1$:

$$\begin{aligned} u_i^{j-1} &:= u_i^j + E_i^j\,\tilde{h}_{j-1}, && i = 1,2,\\ E_i^{p,j-1} &:= E_i^{p,j} + E_i^j\,\tilde{H}_{j-1}^p, && i = 1,2,\\ E_1^{j-1} &:= \tilde{R}^{j-1} + E_1^j\,\tilde{H}_{j-1},\\ E_2^{j-1} &:= \tilde{D}^{j-1} + E_2^j\,\tilde{H}_{j-1}. \end{aligned}$$

Wir erhalten eine äquivalente Umformung des Ausgleichsproblems in ein kondensiertes Problem in den Variablen $(\Delta s_0^y, \Delta p)$

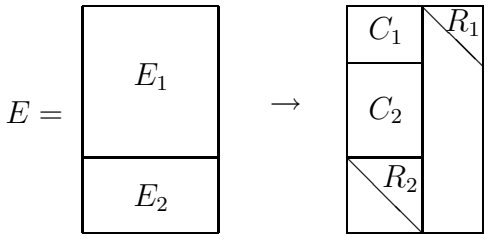
$$\begin{aligned} &\min_{\Delta s_0^y, \Delta p} \|u_1^0 + E_1^0\,\Delta s_0^y + E_1^{p,0}\,\Delta p\|_2^2\\ &\text{so daß } u_2^0 + E_2^0\,\Delta s_0^y + E_2^{p,0}\,\Delta p = 0\,. \end{aligned} \tag{4.12}$$

Zur numerischen Lösung hiervon wird die Matrix $E = \begin{pmatrix} E_1^0 & E_1^{p,0} \\ E_2^0 & E_2^{p,0} \end{pmatrix}$ vollständig auf Dreiecksgestalt gebracht. Dies kann zum Beispiel mit Hilfe von Äquivalenztransformationen nach einem Algorithmus von Bock [Boc87] geschehen. Die Matrix E hat dann folgende Gestalt:



(4.13)

Alternativ können auch nach einem Algorithmus von Stoer [Sto71] orthogonale Transformationen $E_2 = Q_2^l [R_2 0]\, Q_2^r$ angewendet werden. Die Matrix E wird dann auf folgende Gestalt gebracht:



(4.14)

Wir erhalten die Inkremente $(\Delta s_0^y, \Delta p)$ und berechnen sukzessive die Variablen

$$\Delta s_i^y := \tilde{H}_{i-1}\, \Delta s_{i-1}^y + \tilde{H}_{i-1}^p\, \Delta p + \tilde{h}_{i-1},\ i = 1, \ldots, m. \tag{4.15}$$

Anschließend berechnen wir aus Gleichung (4.11) die Variablen Δs_i^z, $i = 0, \ldots, m$.

Eine neue, bessere Approximation ist gegeben durch

$$\nu^{(k+1)} \;=\; \nu^{(k)} + \Delta\nu^{(k)}\,.$$

Das Gauß-Newton-Verfahren hat den Vorteil, daß es lineare Probleme in einem Schritt löst und für nicht allzu nichtlineare Probleme sehr gute Konvergenzeigenschaften besitzt. Probleme gibt es jedoch, wenn der vorgegebene Startwert $\nu^{(0)}$ nicht im lokalen Konvergenzbereich des Verfahrens liegt. Bei den in der Parameterschätzung betrachteten nichtlinearen und komplexen Problemstellungen ist dies sehr oft der Fall. Zur Vergrößerung des Konvergenzbereichs verwendet man eine Dämpfung

$$\nu^{(k+1)} = \nu^{(k)} + \lambda_k\, \Delta\nu^{(k)} \tag{4.16}$$

mit Schrittweite $0 < \lambda_k \leq 1$. Die Berechnung der Schrittweite λ_k kann zum Beispiel mit Hilfe von Liniensuchmethoden erfolgen.

Der Hauptrechenaufwand des oben dargestellten Algorithmus liegt in der Auswertung des Least-Squares-Funktionals und der Nebenbedingungen und deren Ableitungen. Die Ableitungen der Konsistenzbedingungen g nach Anfangswerten s_j und Parametern p können mit numerischen Differenzen oder mit Automatischer Differentiation berechnet werden. Für die Ableitung der Meßfunktionen b, der Gleichungsbedingungen d und der Stetigkeitsbedingungen h benötigt man insbesondere in jedem Diskretisierungsintervall die Ableitung der Lösungstrajektorie $x(t; \tau_j, s_j, p, q)$ des DAE-Systems nach Anfangswerten s_j und Parametern p, worin der Hauptaufwand bei der Berechnung der Ableitungen des nichtlinearen Ausgleichsproblems (4.4)-(4.8) liegt. Für den oben dargestellten Ansatz wird die Ableitung der Lösungstrajektorie des DAE-Systems (4.4) nach allen Anfangswerten und Parametern benötigt. Die effiziente numerische Berechnung dieser Ableitungen mit Hilfe der Techniken der Internen Numerischen Differentiation wird in Kapitel 6 beschrieben.

Bemerkung 4.1.2 (Kollokationsverfahren)
Eine weitere Möglichkeit zur Lösung der unendlich-dimensionalen beschränkten Parameterschätzprobleme ist die Diskretisierung mit Hilfe von Kollokationsverfahren. Dabei wird die Lösung des DAE-Systems mit Hilfe von Basisfunktionen parametrisiert. Kollokationsverfahren für Optimierungsprobleme in ODE/DAE-Systemen wurden voneinander unabhängig von Ascher und Mitarbeitern [ACR81, BA87], Bock und Mitarbeitern ([Boc83], Bär [Bär83], Schulz [Sch90]), von Biegler und Mitarbeitern [CB87, LB89, VB90] und von Renfro et al. [RMA87] entwickelt.

4.2 Optimalitätsbedingungen

Wir schreiben das nichtlineare endlich-dimensionale Least-Squares-Problem verkürzt als

$$\begin{aligned} &\min_{v} \|r(\nu)\|_2^2 \\ &\text{so daß } c(\nu) = 0. \end{aligned} \tag{4.17}$$

Im Vektor ν fassen wir die Optimierungsvariablen (p, s) zusammen, $c : I\!R^{n_\nu} \to I\!R^{n_c}$ enthält die Nebenbedingungen d, die Stetigkeitsbedingungen h und die Konsistenzbedingungen g.

4.2.1 Karush-Kuhn-Tucker-Bedingungen für beschränkte Optimierungsprobleme

Optimalitätsbedingungen für endlich-dimensionale beschränkte Optimierungsprobleme der Form (4.17) wurden von Karush und von Kuhn und Tucker formuliert. Sie werden deshalb in der Regel Karush-Kuhn-Tucker-Bedingungen genannt. Punkte, die die Optimalitätsbedingungen erfüllen, heißen Karush-Kuhn-Tucker-Punkte.

Die Funktionen r und c müssen mindestens einmal stetig differenzierbar sein, für Bedingungen zweiter Ordnung müssen auch die zweiten partiellen Ableitungen stetig sein.

Definition 4.2.1 (Zulässiger Bereich)
Der zulässige Bereich *des Optimierungsproblems (4.17) ist die Menge aller Punkte ν, die die Nebenbedingungen erfüllen*

$$\mathcal{M}_{zul} = \{\nu \in I\!R^{n_\nu} | c(\nu) = 0\}.$$

Definition 4.2.2 (Lokales Minimum)
Ein Punkt $\nu^* \in \mathcal{M}_{zul}$ heißt lokales Minimum, *falls eine Umgebung $\mathcal{U}$ von ν^* existiert, so daß*

$$\|r(\nu^*)\|_2^2 \leq \|r(\nu)\|_2^2 \quad \forall \nu \in \mathcal{U} \cap \mathcal{M}_{zul}. \tag{4.18}$$

Falls die Ungleichung (4.18) strikt ist, so heißt ν^* striktes lokales Minimum.

Definition 4.2.3 (Regulärer Punkt)
Ein Punkt $\nu^* \in \mathcal{M}_{zul}$ heißt regulär, *falls die Vektoren $\nabla c_i(\nu^*), i = 1, \ldots, n_c$ linear unabhängig sind. Diese Bedingung heißt im allgemeinen auch* first-order constraint qualification.

Bei der Formulierung der Optimalitätsbedingungen spielt die *Lagrange-Funktion*

$$\mathcal{L}(\nu, \lambda) = \frac{1}{2}\|r(\nu)\|_2^2 - \lambda^T c(\nu)$$

eine zentrale Rolle. Die Komponenten des Vektors λ heißen dabei *Lagrange-Multiplikatoren.*

Satz 4.2.1 (Notwendige Bedingungen)
Sei $\nu^* \in \mathcal{M}_{zul}$ regulär und lokales Minimum. Dann existiert ein $\lambda \in \mathbb{R}^{n_c}$ mit

i) (Notwendige Bedingung 1. Ordnung)

$$\nabla_\nu \mathcal{L}(\nu^*, \lambda) = 0$$

ii) (Notwendige Bedingung 2. Ordnung)

$$w^T \nabla_{\nu\nu} \mathcal{L}(\nu^*, \lambda)\, w \geq 0$$

$$\forall w \in \{\omega \in \mathbb{R}^{n_\nu} | \nabla c_i(\nu^*)\, \omega = 0, i = 1, \ldots, n_c\}$$

4.2.2 Karush-Kuhn-Tucker-Bedingungen für das Parameterschätzproblem

Schreiben wir das linearisierte Ausgleichsproblem (4.9) in verkürzter Form als

$$\begin{aligned} &\min_{\Delta\nu} \|r(\nu) + J_1(\nu)\, \Delta\nu\|_2^2 \\ &\text{so daß } c(\nu) + J_2(\nu)\, \Delta\nu = 0, \end{aligned} \tag{4.19}$$

so ist es auch das im Punkt ν linearisierte Problem von (4.17).

Satz 4.2.2 (Existenz einer verallgemeinerten Inverse)
Sei das beschränkte linearisierte Ausgleichsproblem (4.9) gegeben und folgende Voraussetzungen seien erfüllt:

$$[CQ]: \quad \text{Rang } J_2 = n_c\, (\leq n_\nu) \qquad \textit{(constraint qualification)} \tag{4.20}$$

$$[PD]: \quad \text{Rang} \begin{pmatrix} J_1 \\ J_2 \end{pmatrix} = n_\nu \qquad \textit{(positive definiteness)} \tag{4.21}$$

$$(\Leftrightarrow \nu^T J_1^T J_1 \nu > 0\, \forall\, \nu \in (\text{Kern } J_2) \backslash \{0\}).$$

Dann gilt:

1) *Es existiert genau ein Karush-Kuhn-Tucker-Punkt (ν^*, λ^*) des beschränkten linearisierten Ausgleichsproblems (4.19) und $\Delta\nu^*$ ist striktes Minimum.*

2) *Es existiert eine lineare Abbildung $J^+ : \mathbb{R}^{n_\nu} \to \mathbb{R}^{n_r+n_c}$, so daß*

$$\Delta\nu^* = -J^+ \begin{pmatrix} r(\nu^*) \\ c(\nu^*) \end{pmatrix}$$

die Lösung des Ausgleichsproblems ist.

3) *Der Lösungsoperator J^+ erfüllt die Bedingung*

$$J^+ J J^+ = J^+$$

und ist somit eine verallgemeinerte Inverse von J.

Bemerkung 4.2.1

1) *(ν^*, λ^*) ist Karush-Kuhn-Tucker-Punkt des nichtlinearen Ausgleichsproblems (4.17) genau dann, wenn $(0, \lambda^*)$ Karush-Kuhn-Tucker-Punkt des linearisierten Ausgleichsproblems (4.19) ist.*

2) *Die notwendigen Bedingungen 1. Ordnung für das linearisierte Problem lauten*

$$\begin{pmatrix} J_1^T J_1 & J_2^T \\ J_2 & 0 \end{pmatrix} \begin{pmatrix} \Delta\nu^* \\ -\lambda^* \end{pmatrix} + \begin{pmatrix} J_1^T r \\ c \end{pmatrix} = 0.$$

Hiermit erhält man eine explizite Darstellung des Lösungsoperators J^+

$$\Delta\nu^* = -\underbrace{\begin{pmatrix} I & 0 \end{pmatrix} \begin{pmatrix} J_1^T J_1 & J_2^T \\ C & 0 \end{pmatrix}^{-1} \begin{pmatrix} J_1^T & 0 \\ 0 & I \end{pmatrix}}_{J^+} \begin{pmatrix} r \\ c \end{pmatrix}.$$

3) *Im Falle eines unbeschränkten Ausgleichsproblems (das heißt $c \equiv 0$) folgt aus der Voraussetzung [PD], daß J_1 Vollrang hat. Die Karush-Kuhn-Tucker-Bedingungen sind dann von der Form*

$$J_1^T J_1 \Delta\nu^* + J_1^T r = 0$$

und wir erhalten

$$\Delta\nu^* = -\underbrace{\left(J_1^T J_1\right)^{-1} J_1^T}_{J^+} r.$$

4.3 Lokale Konvergenz

Der in Bock [Boc87] dargestellte Kontraktionssatz gibt Aussagen über das Konvergenzverhalten des verallgemeinerten Gauß-Newton-Verfahrens (mit Vollschritt, das heißt falls $\lambda_k = 1$ in der Iterationsvorschrift (4.16)).

Satz 4.3.1 (Lokaler Kontraktionssatz)
Sei $J^+(\nu)$ die verallgemeinerte Inverse der Jacobi-Matrix $J(\nu)$ des nichtlinearen Ausgleichsproblems (4.5) bis (4.8) mit rechter Seite $F(\nu) = (r(\nu)^T\, r(\nu), c(\nu)^T)^T \in \mathcal{C}^1(D)$. Ferner seien die folgenden Lipschitz-Bedingungen für alle $t \in [0,1]$ und alle $\nu, w, z \in D$ mit $\nu - w = J(\nu)^+ F(\nu)$ erfüllt:

i) $\|J^+(w)(J(\nu + \tau(w-\nu)) - J(\nu))\,(w-\nu)\| \le \omega^{(k)}\,\tau\,\|w-\nu\|^2, \quad \omega^{(k)} \le \omega < \infty$

ii) $\|(J^+(z) - J^+(\nu))\,(F(\nu) - J(\nu)J^+(\nu)F(\nu)\| \le \kappa^{(k)}\|z-\nu\|, \quad \kappa^{(k)} \le \kappa < 1,$

iii) für alle $\nu^0 \in D$ gelte

$$\delta_0 := \frac{\omega^0}{2}\|\Delta\nu^0\| + \kappa^0 < 1,\ \Delta\nu^0 = \|J^+(\nu^0)F(\nu^0)\| \tag{4.22}$$

iv) und die Kugel $D_0 := S_r(\nu^0)$ um ν^0 mit Radius $r = \frac{\|\Delta\nu^0\|}{1-\delta_0}$ liege in D.

Dann folgt:

1. *Die Iteration $\nu^{(k+1)} = \nu^{(k)} + \Delta\nu^{(k)}$, $\Delta\nu^{(k)} = -J^+(\nu^{(k)})\,F(\nu^{(k)})$, ist wohldefiniert und bleibt in D_0.*
2. *Es existiert eine Nullstelle $\nu^* \in D_0$ gegen die $\nu^{(k)}$ konvergiert mit $J^+(\nu^*)\,F(\nu^*) = 0$.*
3. *Die Folge $\nu^{(k)}$ konvergiert mindestens linear mit*

$$\|\Delta\nu^{(k)}\| \le \left(\frac{\omega^{(k-1)}}{2}\|\Delta\nu^{(k-1)}\| + \kappa^{(k-1)}\right)\|\Delta\nu^{(k-1)}\| \le \|\Delta\nu^{(k-1)}\|$$

4. *und für die k-te Iterierte gilt die a priori-Abschätzung*

$$\|\nu^{(k)} - \nu^*\| \le \|\Delta\nu^0\|\frac{\delta_0^k}{1-\delta_0}\,. \tag{4.23}$$

Die Lipschitz-Konstante ω beschreibt wie in den Sätzen 2.3.1 und 3.1.1 die Nichtlinearität von J. Da sie ein Maß für den Gültigkeitsbereich der Linearisierung ist, basiert die Bestimmung der Schrittweite λ_k in (4.16) für das gedämpfte Gauß-Newton-Verfahren auf Schätzungen von ω.

Die Inkompatibilitätskonstante κ beschreibt in diesem Zusammenhang die Identifizierbarkeit des nichtlinearen Modells aus fehlerbehafteten Meßdaten. Bock zeigt in [Boc87], daß für $\kappa < 1$ der Fixpunkt ν^* ein striktes lokales Minimum ist und somit stabil gegen Störungen. Für $\kappa \ge 1$ existieren Störungen der Meßdaten in der Größenordnung der Meßfehler, so daß der Fixpunkt zwar stationär (d.h. $J^+(\nu^*)\,F(\nu^*) = 0$), aber kein Minimum mehr ist.

4.4 Reduzierter Ansatz

Der in Abschnitt 4.1.2 dargestellte Algorithmus zur Lösung des linearisierten Ausgleichsproblems (4.9) erfordert insbesondere die Berechnung der Matrizen $H_j^{s_j}$ und H_j^p, $j = 0, \ldots, m-1$, das heißt die Berechnung der Ableitungen der Lösungstrajektorie des DAE-Systems nach allen Anfangswerten und Parametern. Mit wachsender Anzahl der Zustandsvariablen des DAE-Systems wächst der Aufwand zur numerischen Berechnung des Parameterschätzproblems drastisch.

Unter der Voraussetzung separabler Nebenbedingungen hat Schlöder [Sch88] ein spezielles Mehrzielverfahren entwickelt, dessen Effizienz auf dem geschickten Ausnutzen der Gleichungsnebenbedingungen beruht. Diese Vorgehensweise wurde im verallgemeinerten Gauß-Newton-Verfahren FIXFIT implementiert. Die grundlegende Idee besteht darin, die Berechnung der einzelnen Blöcke der Mehrzielmatrix J und der anschließenden Elimination effizient zu koppeln. Durch fortgesetztes Bilden von Richtungsableitungen liegt der Aufwand zur Berechnung von Richtungen der Variationsdifferentialgleichung in der Dimension der Freiheitsgrade des Systems. Für das Least-Squares-Funktional r und die Konsistenz- und Stetigkeitsbedingungen ist die Separabilität per definitionem erfüllt. Der Übersichtlichkeit halber nehmen wir für die Gleichungsnebenbedingungen Separabilität in jedem Mehrzielknoten an

$$\bar{d}(s_0, \ldots, s_m, p) =: \begin{pmatrix} \hat{d}_0(s_0, p) \\ \vdots \\ \hat{d}_m(s_m, p) \end{pmatrix} \tag{4.24}$$

und somit

$$D(s_0, \ldots, s_m, p) =: \begin{pmatrix} \hat{D}_0(s_0, p) & & \\ & \ddots & \\ & & \hat{D}_m(s_m, p) \end{pmatrix} \tag{4.25}$$

und untersuchen das Ausnutzen der Gleichungsbedingungen im Zusammenhang mit Parameterschätzproblemen bei DAE-Systemen.

Wir nehmen O.B.d.A. an, daß die zu $\begin{pmatrix} \hat{D}_0 \\ G_0 \end{pmatrix}$ gehörigen Zeilen des Ausgleichsproblems Vollrang in $\Delta s_0 = \begin{pmatrix} \Delta s_0^y \\ \Delta s_0^z \end{pmatrix}$ haben. Wir partitionieren diese Zeilen so, daß die Matrix $\begin{pmatrix} \hat{D}_0^{s_{02}^y} & \hat{D}_0^{s_0^z} \\ G_0^{s_{02}^y} & G_0^{s_0^z} \end{pmatrix}$ quadratisch und regulär ist ($G_0^{s_0^z}$ ist aufgrund der Index-1-Bedingung regulär, für die differentiellen Variablen muß eventuell Δs_0^y zu $({\Delta s_{01}^y}^T, {\Delta s_{02}^y}^T)^T$ umgeordnet werden). Aus den Zeilen

$$\begin{aligned} \hat{D}_0^{s_{01}^y} \Delta s_{01}^y + \hat{D}_0^{s_{02}^y} \Delta s_{02}^y + \hat{D}_0^{s_0^z} \Delta s_0^z + \hat{D}_0^p \Delta p + \hat{d}_0(s_0, p) = 0 \\ G_0^{s_{01}^y} \Delta s_{01}^y + G_0^{s_{02}^y} \Delta s_{02}^y + G_0^{s_0^z} \Delta s_0^z + G_0^p \Delta p + g_0(s_0, p) = 0 \end{aligned} \tag{4.26}$$

eliminieren wir formal die $n_{elim} = n_{y_2} + n_z$ Variablen $({\Delta s_{02}^y}^T, {\Delta s_0^z}^T)$ und erhalten

$$\begin{pmatrix} \Delta s_{02}^y \\ \Delta s_0^z \end{pmatrix} = - \begin{pmatrix} \hat{D}_0^{s_{02}^y} & \hat{D}_0^{s_0^z} \\ G_0^{s_{02}^y} & G_0^{s_0^z} \end{pmatrix}^{-1} \left(\begin{pmatrix} \hat{D}_0^{s_{01}^y} \\ G_0^{s_{01}^y} \end{pmatrix} \Delta s_{01}^y + \begin{pmatrix} \hat{D}_0^p \\ G_0^p \end{pmatrix} \Delta p + \begin{pmatrix} \hat{d}_0 \\ g_0 \end{pmatrix} \right). \tag{4.27}$$

Wir setzen

$$
\begin{aligned}
{}^{-1}H_E &:= \begin{pmatrix} & I_{n_{y_1}} \\ -\begin{pmatrix} \hat{D}_0^{s_{02}^y} & \hat{D}_0^{s_0^z} \\ G_0^{s_{02}^y} & G_0^{s_0^z} \end{pmatrix}^{-1} & \begin{pmatrix} \hat{D}_0^{s_{01}^y} \\ G_0^{s_{01}^y} \end{pmatrix} \end{pmatrix}, \\
{}^{-1}H_P &:= \begin{pmatrix} & 0_{n_{y_1} \times n_p} \\ -\begin{pmatrix} \hat{D}_0^{s_{02}^y} & \hat{D}_0^{s_0^z} \\ G_0^{s_{02}^y} & G_0^{s_0^z} \end{pmatrix}^{-1} & \begin{pmatrix} \hat{D}_0^{p} \\ G_0^{p} \end{pmatrix} \end{pmatrix}, \\
{}^{-1}H_R &:= \begin{pmatrix} & 0_{n_{y_1} \times 1} \\ -\begin{pmatrix} \hat{D}_0^{s_{02}^y} & \hat{D}_0^{s_0^z} \\ G_0^{s_{02}^y} & G_0^{s_0^z} \end{pmatrix}^{-1} & \begin{pmatrix} \hat{d}_0 \\ g_0 \end{pmatrix} \end{pmatrix}.
\end{aligned}
$$

Die Spalten von $\begin{pmatrix} {}^{-1}H_E & {}^{-1}H_P \\ 0_{n_p \times n_{y_1}} & I_{n_p} \end{pmatrix}$ bilden eine Basis des Nullraums von (4.26).

Zusätzlich definieren wir

$$
\begin{aligned}
\hat{E}_{11}^0 &:= R_0 \cdot {}^{-1}H_E \\
\hat{E}_{21}^0 &:= \begin{pmatrix} \hat{D}_0 \\ G_0 \end{pmatrix} \cdot {}^{-1}H_E \\
\hat{E}_{1p}^0 &:= R_0 \cdot {}^{-1}H_P + R^p \\
\hat{E}_{2p}^0 &:= \begin{pmatrix} \hat{D}_0 \\ G_0 \end{pmatrix} \cdot {}^{-1}H_P + \begin{pmatrix} \hat{D}_0^p \\ G_0^p \end{pmatrix} \\
\hat{u}_1^0 &:= R_0 \cdot {}^{-1}H_R + r_0 \\
\hat{u}_2^0 &:= \begin{pmatrix} \hat{D}_0 \\ G_0 \end{pmatrix} \cdot {}^{-1}H_R + \begin{pmatrix} \hat{d}_0 \\ g_0 \end{pmatrix}.
\end{aligned}
$$

Dann berechnen wir rekursiv für $j = 1, \ldots, m$

$$
\begin{aligned}
\hat{E}_{11}^j &:= \hat{E}_{11}^{j-1} + R_j \cdot {}^{j-1}H_E \\
\hat{E}_{21}^j &:= \hat{E}_{21}^{j-1} + \begin{pmatrix} \hat{D}_j \\ G_j \end{pmatrix} \cdot {}^{j-1}H_E \\
\hat{E}_{1p}^j &:= \hat{E}_{1p}^{j-1} + R_j \cdot {}^{j-1}H_P + R_j^p \\
\hat{E}_{2p}^j &:= \hat{E}_{2p}^{j-1} + \begin{pmatrix} \hat{D}_j \\ G_j \end{pmatrix} \cdot {}^{j-1}H_P + \begin{pmatrix} \hat{D}_j^p \\ G_j^p \end{pmatrix} \\
\hat{u}_1^j &:= \hat{u}_1^{j-1} + R_j \cdot {}^{j-1}H_R + r_j \\
\hat{u}_2^j &:= \hat{u}_2^{j-1} + \begin{pmatrix} \hat{D}_j \\ G_j \end{pmatrix} \cdot {}^{j-1}H_R + \begin{pmatrix} \hat{d}_j \\ g_j \end{pmatrix}.
\end{aligned}
$$

Mit $\tilde{E}_{i1} := \hat{E}_{i1}^m$, $\tilde{E}_i^p := \hat{E}_{ip}^m$, $\tilde{u}_i := \hat{u}_i^m$, $i = 1, 2$, erhalten wir das reduzierte kondensierte

System

$$\begin{aligned} &\min_{\Delta s_{01}^y, \Delta p} \|\tilde{u}_1 + \tilde{E}_{11}^0 \, \Delta s_{01}^y + \tilde{E}_1^p \, \Delta p\|_2^2 \\ &\text{so daß } \; \tilde{u}_2 + \tilde{E}_{21}^0 \, \Delta s_{01}^y + \tilde{E}_2^p \, \Delta p = 0 \, . \end{aligned} \tag{4.28}$$

Dies kann – analog zum kondensierten System (4.12) für das allgemeine Problem – mit Hilfe von Äquivalenz- (4.13) oder Orthogonaltransformationen (4.14) gelöst werden. Nach der Berechnung der Inkremente $({\Delta s_{01}^y}^T, \Delta p^T)$ lassen sich die weiteren Inkremente $({\Delta s_{02}^y}^T, \Delta s_0^{zT}, \Delta s_1, \ldots, \Delta s_m)$ berechnen durch

$$\Delta s_j := {}^{j-1}H_E \, \Delta s_{01}^y + {}^{j-1}H_P \, \Delta p + {}^{j-1}H_R, \; j = 0, \ldots, m. \tag{4.29}$$

Die Ausdrücke ${}^{j}H_\kappa$, $\kappa \in \{E, P, R\}$, $j \geq 0$, werden rekursiv berechnet

$$\begin{aligned} {}^{j}H_E &:= H_j \cdot {}^{j-1}H_E \\ {}^{j}H_P &:= H_j \cdot {}^{j-1}H_P + H_j^p \\ {}^{j}H_R &:= H_j \cdot {}^{j-1}H_R + h_j. \end{aligned}$$

Die Terme $\bar{H}_j := H_j \cdot {}^{j-1}H_\kappa$, $j = 1, \ldots, m-1$, $\kappa \in \{E, P, R\}$, können nun über Richtungsableitungen berechnet werden. Im Gegensatz zu $n_x + n_p$ Richtungen im allgemeinen Fall, müssen hier nur noch $(n_x - n_{elim}) + n_p + 1$ Richtungen berechnet werden. Bei festen Anfangswerten ist die Anzahl der Richtungen sogar unabhängig von der Dimension n_x der Zustandsvariablen des DAE-Systems.

4.5 Mehrfachexperimentprobleme

Bei dem komplexen und nichtlinearen Zusammenhang zwischen Meßdaten und unbekannten Parametern ist man oft nicht in der Lage, alle Parameter aus einem Experiment zu schätzen. Meist müssen viele Experimente mit unterschiedlichen Einstellungen durchgeführt werden, um sie mit der gewünschten Genauigkeit zu erhalten.

Für diese Problemstellung wurden von Schlöder [Sch88] und von Schwerin [vS98] Algorithmen entwickelt, die die speziellen Strukturen, die sich aufgrund des Mehrfachexperiment-Ansatzes ergeben, ausnutzen. Wir erhalten ein Optimierungsproblem ähnlich zu (4.1) - (4.3) für N_{ex} Experimente:

$$\min_{p,x} \sum_{l=1}^{N_{ex}} \sum_{i,j} {}^{l}w_{ij} \cdot \frac{({}^{l}\eta_{ij} - {}^{l}b_i({}^{l}t_j, {}^{l}x({}^{l}t_j), p, {}^{l}q))^2}{{}^{l}\sigma_{ij}{}^2} \, . \tag{4.30}$$

so daß für jedes Experiment $l, l = 1, \ldots, N_{ex}$, gilt:

- die Zustandsvariablen ${}^{l}x = ({}^{l}y, {}^{l}z)$ lösen das DAE-System

$$\begin{aligned} {}^{l}A({}^{l}t, {}^{l}y, {}^{l}z, p, {}^{l}q, {}^{l}u) \; {}^{l}\dot{y} &= {}^{l}f({}^{l}t, {}^{l}y, {}^{l}z, p, {}^{l}q, {}^{l}u) && \text{(4.31a)} \\ 0 &= {}^{l}g({}^{l}t, {}^{l}y, {}^{l}z, p, {}^{l}q, {}^{l}u) && \text{(4.31b)} \end{aligned}$$

- und erfüllen die zusätzlichen Nebenbedingungen

$$ {}^{l}d({}^{l}x({}^{l}t_0), \ldots, {}^{l}x({}^{l}t_{f_l}), p, {}^{l}q) = 0\,. \tag{4.32} $$

Das DAE-System kann dabei von Experiment zu Experiment variieren, genauso wie die Meßfunktionen ${}^{l}b$ und die zugehörigen Varianzen ${}^{l}\sigma^2$. Die numerische Lösung des Parameterschätzproblems erfolgt wiederum mit einem direkten Ansatz. Für jedes Experiment wird der Beobachtungszeitraum $[{}^{l}t_0, {}^{l}t_{f_l}]$ diskretisiert in

$$ {}^{l}t_0 = {}^{l}\tau_0 < \cdots < {}^{l}\tau_{m_l} = {}^{l}t_{f_l}. $$

Zur Parametrisierung der unendlich-dimensionalen Lösung des DAE-Systems führen wir wiederum Variablen ${}^{l}s_j$ an den Stützstellen ein. Auf jedem Teilintervall $[{}^{l}\tau_j, {}^{l}\tau_{j+1}]$ lösen wir eine relaxierte Form des DAE-Systems (4.31) für Experiment l und fügen die Konsistenzbedingungen

$$ {}^{l}g({}^{l}\tau_j, {}^{l}s_j, p, {}^{l}q, {}^{l}u) = 0 \tag{4.33} $$

zu den Nebenbedingungen hinzu. Wir lösen das resultierende endlich-dimensionale Optimierungsproblem mit einem verallgemeinerten Gauß-Newton-Verfahren ähnlich zum Einfachexperimentfall. Pro Experiment führen wir eine Kondensierung wie in den Abschnitten 4.1.2 oder 4.4 beschrieben durch. Wir erhalten ein strukturiertes kondensiertes System in den Variablen $({}^{1}\nu_0, \ldots, {}^{N_{ex}}\nu_0, p)$ mit Jacobi-Matrix

$$ J_{MFE} = \begin{pmatrix} E_1^{\nu} & & 0 & E_1^{p} \\ & \ddots & & \vdots \\ 0 & & E_{N_{ex}}^{\nu} & E_{N_{ex}}^{p} \end{pmatrix} \tag{4.34} $$

und neuer rechter Seite

$$ u_{MFE} = -\begin{pmatrix} \bar{u}_1 \\ \vdots \\ \bar{u}_{N_{ex}} \end{pmatrix}, \quad \bar{u}_l = \begin{pmatrix} {}^{l}\bar{u}_1 \\ {}^{l}\bar{u}_2 \end{pmatrix}, \, l = 1, \ldots, N_{ex}. $$

Dabei gilt

$$ {}^{l}\nu_0 = {}^{l}s_0^{y}, \quad E_l^{\nu} = \begin{pmatrix} {}^{l}E_1^0 \\ {}^{l}E_2^0 \end{pmatrix}, \quad E_l^{p} = \begin{pmatrix} {}^{l}E_1^{p,0} \\ {}^{l}E_2^{p,0} \end{pmatrix} \text{ und } {}^{l}\bar{u}_i = u_i^0, \, i = 1, 2, \, l = 1, \ldots, N_{ex}, $$

für den allgemeinen Ansatz, wie in Abschnitt 4.1.2 beschrieben, beziehungsweise

$$ {}^{l}\nu_0 = {}^{l}s_{01}^{y}, \quad E_l^{\nu} = \begin{pmatrix} {}^{l}\tilde{E}_{11}^0 \\ {}^{l}\tilde{E}_{21}^0 \end{pmatrix}, \quad E_l^{p} = \begin{pmatrix} {}^{l}\tilde{E}_1^{p} \\ {}^{l}\tilde{E}_2^{p} \end{pmatrix} \text{ und } {}^{l}\bar{u}_i = \hat{u}_i^m, \, i = 1, 2, \, l = 1, \ldots, N_{ex}, $$

für den reduzierten Ansatz, wie in Abschnitt 4.4 beschrieben.

Zunächst werden in jedem Experiment die zu den lokalen Variablen ${}^{l}\nu_0$ gehörigen Blöcke auf obere Dreiecksgestalt gebracht. Die restlichen zu den Parametern p gehörigen Zeilen

werden nach Gleichungs- und Ausgleichsbedingungen geordnet und (zum Beispiel mit Hilfe orthogonaler Transformationen) auf obere Dreiecksgestalt gebracht.

Daraus können zunächst die Inkremente Δp berechnet werden und im Anschluß die Inkremente ${}^l\nu_0$, $l = 1, \ldots, N_{ex}$. Mit Hilfe der Rückwärtsrekursionen (4.15) und (4.11) für den allgemeinen Ansatz beziehungsweise (4.29) für den reduzierten Ansatz werden schließlich die Variablen $(\Delta\, {}^l s_0, \ldots, \Delta\, {}^l s_{m_l})$ berechnet.

Bemerkung 4.5.1 (Parallelisierbarkeit)
Da die Integration und Ableitungsgenerierung im nichtreduzierten Ansatz in den Mehrzielintervallen voneinander unabhängig berechnet werden kann, läßt eine parallele Variante der Methoden einen großen SpeedUp erwarten. Für die Parameterschätzung bei ODE-Systemen haben Gallitzendörfer und Bock für den Einfachexperimentfall [GB94, Gal97, ZBGS96] und zusammen mit von Schwerin für den Mehrfachexperimentfall [Gal97, vS98] Verfahren entwickelt, die die Strukturen aufgrund des Mehrzielansatzes und im letzteren Fall zusätzlich aufgrund des Mehrfachexperimentansatzes ausnutzen. Für Parameterschätzprobleme bei nicht-steifen Systemen läßt sich auch die Integration und Ableitungsgenerierung zusätzlich einfach parallelisieren.

4.6 Sensitivitätsanalyse der geschätzten Parameter

Da die Meßdaten η_{ij} Zufallsvariablen sind, sind es auch die geschätzten Parameter p und die zusätzlich eingeführten Variablen s. Diese sind in erster Näherung wiederum normalverteilt

$$(p, s) \sim \mathcal{N}\left((p^*, s^*), C_{(p^*,s^*)}\right)$$

mit dem (unbekannten) wahren Wert (p^*, s^*) als Erwartungswert und Kovarianzmatrix $C_{(p^*,s^*)}$. Die exakte Berechnung des nichtlinearen Konfidenzgebiets um den wahren Wert (p^*, s^*) ist in der Praxis nicht möglich, da zum einen der wahre Wert nicht bekannt ist und zum anderen die Berechnung des nichtlinearen Konfidenzgebiets zu aufwendig ist.

Die (näherungsweise) Kovarianzmatrix

$$C = C(\hat{p}, \hat{s}) = J^+ \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} J^{+T} \tag{4.35}$$

ist eine Approximation des nichtlinearen Konfidenzgebiets. Sie kennzeichnet ein Konfidenzellipsoid um die geschätzten Parameter $(\hat{p}, \hat{s})$. Der besondere Vorteil bei der Mehrzielparametrisierung der Lösung des DAE-Systems ist, daß die Kovarianzmatrix nicht nur für die geschätzten Parameter $\hat{p}$, sondern auch für die Lösung des DAE-Systems an bestimmten Zeitpunkten (oder Ortspunkten) berechnet werden kann.

Das Konfidenzellipsoid des im Lösungspunkt $(\hat{p}, \hat{s})$ linearisierten beschränkten Ausgleichsproblems (4.19) wird beschrieben durch

$$G_L(\alpha; \hat{p}, \hat{s}, q, u, w) = \left\{ (p, s) \;\middle|\; \bar{d}(\hat{s}, \hat{p}) + D \begin{pmatrix} p - \hat{p} \\ s - \hat{s} \end{pmatrix} = 0, \right.$$

$$\left\| r(\hat{s},\hat{p}) + R \begin{pmatrix} p-\hat{p} \\ s-\hat{s} \end{pmatrix} \right\|_2^2 - \left\| r(\hat{s},\hat{p}) \right\|_2^2 \leq \gamma^2(\alpha) \Bigg\}.$$

Der Wahrscheinlichkeitsfaktor $\gamma^2(\alpha)$ ist gegeben durch das Quantil der χ^2-Verteilung zum Wert $\alpha \in [0,1]$ mit $l_1 - l_2$ Freiheitsgraden

$$\gamma^2(\alpha) = \chi^2_{1-\alpha}(l_1 - l_2),$$

wobei $l_1 = n_\nu = n_p + n_s$ die Dimension von (p,s) ist und l_2 die Dimension der Nebenbedingungen c.

Falls die Standardabweichungen der Meßfehler nur bis auf einen gemeinsamen Faktor β bekannt sind, ist der Wahrscheinlichkeitsfaktor $\gamma^2(\alpha)$ gegeben durch

$$\gamma^2(\alpha) = b^2(\hat{p},\hat{s}) \cdot (l_1 - l_2) \cdot F_{1-\alpha}(l_1 - l_2, m - (l_1 - l_2)).$$

Dabei bezeichnet m die Anzahl der Messungen, $F_{1-\alpha}$ das Quantil der F-Verteilung mit $l_1 - l_2$ und $m - (l_1 - l_2)$ Freiheitsgraden und $b(\hat{p},\hat{s})$ eine unabhängige Schätzung für den gemeinsamen Faktor (siehe auch Bock [Boc87])

$$b^2(\hat{p},\hat{s}) := \frac{\|r(\hat{p},\hat{s})\|_2^2}{l_1 - l_2}.$$

Die Konfidenzellipsoide $G_L(\alpha;\hat{p},\hat{s})$ werden von einem Quader mit Seitenlängen $2\,\theta_i$ eingeschlossen

$$G_L(\alpha;\hat{p},\hat{s}) \subset \otimes_{i=1}^{l_1} [(\hat{p},\hat{s})_i - \theta_i, (\hat{p},\hat{s})_i + \theta_i]. \tag{4.36}$$

wobei $\theta_i = \gamma(\alpha) \cdot \sqrt{C_{ii}}$ und $\sqrt{C_{ii}}$ die Wurzel des i-ten Diagonalelements der Kovarianzmatrix beschreibt. Hiermit kann (näherungsweise) die statistische Güte der geschätzten Parameter und der Trajektorien, besonders an Stellen, an denen keine Meßdaten erhoben werden können, beurteilt werden: Je kleiner die Seitenlänge θ_i des Quaders (bei gegebenem α), desto besser sind die Parameter bestimmt.

Bemerkung 4.6.1 (Berechnung der Kovarianzmatrix)
Die im nächsten Kapitel betrachteten Versuchsplanungsprobleme basieren auf einem Funktional auf der (näherungsweisen) Kovarianzmatrix C, die wiederum von der Jacobi-Matrix J des zugrundeliegenden Parameterschätzproblems abhängt. Da der numerische Aufwand in der Versuchsplanung unter anderem stark von der Dimension der Kovarianzmatrix abhängt, kann man analog zum reduzierten Ansatz den Teil der durch die Gleichungsbedingungen festgelegten Variablen von s und eventuell auch von p eliminieren und eine Kovarianzmatrix (beziehungsweise zunächst die zugehörige Jacobi-Matrix) zum Beispiel nur noch in der Dimension der Freiheitsgrade des Systems – oder allgemeiner in allen Variablen, an deren statistischer Güte man interessiert ist – aufstellen. Die zugehörige Jacobi-Matrix J kann zudem für ein Parameterschätzproblem mit Meßdaten aus einem einzelnen oder aus mehreren Experimenten (siehe (4.34)) aufgestellt werden.

Die effiziente numerische Berechnung der Kovarianzmatrix wird in Abschnitt 5.4.2 zur Versuchsplanung im Zusammenhang mit der Berechnung der Ableitungen der Kovarianzmatrix näher beschrieben. Ist man nur an den gemeinsamen Konfidenzintervallen interessiert, so benötigt man nur die Diagonalelemente der Kovarianzmatrix. Falls man eine Variante des oben dargestellten verallgemeinerten Gauß-Newton-Verfahrens verwendet, so ist die verallgemeinerte Inverse aus dem letzten Iterationsschritt eine Näherung für $J^+(\hat{p}, \hat{s})$. *Die Berechnung der Diagonalelemente der Kovarianzmatrix kann mit geringem Mehraufwand aus den bereits vorhandenen Zerlegungen der verallgemeinerten Inversen berechnet werden.*

Kapitel 5

Optimal-Steuerungsprobleme zur Versuchsplanung

Im vorhergehenden Kapitel wurde die numerische Behandlung von Parameterschätzproblemen bei DAE-Systemen und die statistische Güte der geschätzten Parameter – unter der Voraussetzung normalverteilter Meßfehler – untersucht. Zu unterschiedlichen Experimentauslegungen können die geschätzten Parameter statistisch sehr weit um den wahren Parametersatz streuen oder auch nur in einem kleinen Bereich. Sind die zu schätzenden Parameter noch nicht genau bestimmt, müssen weitere, oft kostspielige und zeitintensive Experimente durchgeführt werden. Um diesen Aufwand auf ein Minimum zu reduzieren und die statistische Güte der zu schätzenden Parameter zu maximieren, verwenden wir Verfahren der nichtlinearen optimalen Versuchsplanung. Das Ziel ist die optimale Auslegung von Experimenten und die Auswahl von Messungen, so daß die statistische Güte der aus den zugehörgien Meßdaten geschätzten Parameter maximal ist.

Die Approximation der Kovarianzmatrix C (siehe Abschnitt 4.6) gibt Aussagen über die statistische Güte der geschätzten Parameter. Analog zu den Methoden zur Versuchsplanung bei nichtlinearen Regressionsmodellen minimieren wir eine Funktion auf der Kovarianzmatrix. Dabei werden nicht nur die Varianzen und Kovarianzen der zu schätzenden Parameter minimiert, sondern auch von den bei der Parametrisierung der Lösung der DAE-Systeme eingeführten Variablen s. Dies erlaubt die Maximierung der statistischen Güte nicht nur von den zu schätzenden Parametern sondern auch von wichtigen Komponenten der Lösungstrajektorie an Stellen, an denen keine Messungen vorliegen beziehungsweise nicht durchgeführt werden können.

Der zugrundeliegende Prozeß wird – wie bei der Parameterschätzung auch – durch ein DAE-System beschrieben. Wir erhalten ein nichtlineares zustandsbeschränktes Optimal-Steuerungsproblem. Zur Lösung hierfür wählen wir einen direkten Ansatz und überführen das unendlich-dimensionale Problem in ein endlich-dimensionales gleichungs- und ungleichungsbeschränktes Optimierungsproblem. Dieses lösen wir mit Verfahren der sequentiellen quadratischen Programmierung (SQP).

Im folgenden stellen wir zunächst die Formulierung des Versuchsplanungs-Optimie-

rungsproblems und ihrer Einflußgrößen dar und zeigen die numerische Behandlung mit einem direkten Ansatz. Eine besondere Schwierigkeit stellt dabei das unkonventionelle Zielfunktional dar, das eine Funktion auf der Kovarianzmatrix C aus Abschnitt 4.6 und somit implizit eine Funktion auf der Jacobi-Matrix J des zugrundeliegenden Parameterschätzproblems ist. Das Zielfunktional enthält bereits erste Ableitungen der Lösungstrajektorie des DAE-Systems nach den Parametern p und Parametrisierungsvariablen s. Zur numerischen Lösung des Optimierungsproblems mit Verfahren vom Newton-Typ müssen zusätzlich zweite gemischte Ableitungen der Trajektorie nach Parametern, Anfangswerten und Kontrollgrößen bereitgestellt werden. Zur Berechnung der benötigten Ableitungen des Zielfunktionals und der Nebenbedingungen wird eine Mischung aus semi-analytischen Formeln und Automatischer Differentiation verwendet. Die ersten und zweiten Ableitungen der Lösungstrajektorie des DAE-Systems werden mit Techniken der Internen Numerischen Differentiation und mit Automatischer Differentiation mit der von der Optimierung geforderten Genauigkeit berechnet.

Theoretische Untersuchungen zur optimalen Versuchsplanung bei linearen und auch nichtlinearen Regressionsmodellen finden sich schon relativ früh, so zum Beispiel die Übersichtsartikel von Draper und Hunter [DH66], von Kiefer [Kie61], Kiefer und Wolfowitz [KW59] und das umfassende, eher theoretisch orientierte Buch von Fedorov [Fed72].

In der Praxis ist die Versuchsplanung für Modelle, in denen die Parameter linear in das Modell eingehen, heutzutage schon weit verbreitet. Falls der Prozeß durch ein Modell approximiert wird, in das auch die das Experiment bestimmenden Einflußgrößen linear eingehen – oder zumindest nur eine Kopplung durch Polynome endlichen Grades auftritt –, werden häufig faktorielle Designs beziehungsweise auch fraktionelle faktorielle Designs verwendet (siehe z. B. die Bücher von Box et al. [BHH78] und Atkinson und Donev [AD92]).

Falls die Einflußgrößen nichtlinear in das Modell eingehen, verwendet man in der Regel die klassischen Gütekriterien aus der Statistik (Determinante, Spur, größter Eigenwert). Einen Überblick hierzu geben unter anderem die Bücher von Atkinson und Donev [AD92], von Pukelsheim [Puk93] sowie die Artikel von Draper und Pukelsheim [DP96] und von Atkinson [Atk96].

In vielen Fällen wird für nichtlineare Regressionsmodelle eine Approximation der Kovarianzmatrix beziehungsweise der Informationsmatrix analytisch berechnet und darauf zum Beispiel das Determinanten-Kriterium angewendet. Siehe hierzu die Arbeiten von Reilly et al. [RBB+77], Qureshi et al. [QNG80], Rasch [Ras90], Dovíc et al. [DRAD94] und von Rudolph und Herrendörfer [RH95].

Zur Versuchsplanung bei etwas komplexeren dynamischen Prozessen, bei denen das zugrundeliegende Problem durch ein System von gewöhnlichen oder differentiell-algebraischen Gleichungen modelliert wurde, existieren nur wenige Arbeiten. Oinas et al. [OTH92] untersuchten Versuchsplanungsprobleme für ODE- und PDE-Modelle bei mehrphasigen Reaktoren. Die Versuchspläne wurden dabei mit faktoriellen Designs erstellt, wie sie normalerweise nur in der linearen Versuchsplanung üblich sind, bei denen die Parameter linear in das Modell eingehen und die Einflußgrößen nur über Polynome

untereinander gekoppelt sind. Baltes et al. [BSSR94] untersuchten Versuchsplanungsprobleme bei Langzeitselektionsexperimenten. Die Modellierung des Prozesses führt auf ein gewöhnliches Differentialgleichungssystem. Die Optimierung erfolgte mit einem Simplex-Verfahren nach Nelder und Mead [NM64] als Black-Box-Ansatz.

Lohmann et al. [LBS92, Loh93] untersuchten die optimale Selektion von Messungen unter gegebenen Versuchsbedingungen zur Maximierung der statistischen Güte der zu schätzenden Parameter. Die Jacobi-Matrix des zugrundeliegenden beschränkten Parameterschätzproblems ist dabei konstant und muß nur einmal zu Beginn der Optimierung berechnet werden. Hilf [Hil96] untersuchte Versuchsplanungsprobleme zur Roboterkalibrierung. Die von ihm betrachteten Modelle sind nicht steif, so daß für die ersten und zweiten Ableitungen im expliziten Verfahren ein Gesamtsystem aus DAE-Gleichungen und Sensitivitätsgleichungen gelöst werden kann. Die Ableitungen der Modellfunktionen des DAE-Systems konnten analytisch bereitgestellt werden. Außerdem sind die von ihm betrachteten Parameterschätzprobleme unbeschränkt, was zu einer einfacheren Berechnung der Kovarianzmatrix und insbesondere ihrer Ableitungen führt. Raum [Rau97] leitete effiziente Techniken zur Berechnung der benötigten Ableitungen des komplexen Zielfunktionals für Versuchsplanungsprobleme bei gewöhnlichen Differentialgleichungen her. Die zugrundeliegenden Parameterschätzprobleme können dabei beschränkt oder unbeschränkt sein.

5.1 Problemformulierung

Die optimale Versuchsplanung hat zum Ziel, Experimente und Messungen so auszulegen, daß die aus den Meßdaten zu schätzenden Parameter – und eventuell noch zusätzliche unbekannte Variablen – maximale statistische Güte haben. Ein Maß für die statistische Güte der zu schätzenden Größen liefern Funktionen auf der Kovarianzmatrix C. Wir verwenden unter anderem die aus der Versuchsplanung für nichtlineare Regressionsmodelle bekannten Gütekriterien zur Maximierung der statistischen Güte. Die unendlich-dimensionale Lösungstrajektorie des DAE-Systems wird dabei – wie bei der numerischen Behandlung der Parameterschätzprobleme auch – durch die endliche Anzahl von Variablen s parametrisiert. Da es in der Regel nicht ausreicht, die Parameter aus Daten eines Experiments mit der geforderten Genauigkeit zu schätzen und somit mehrere Experimente gleichzeitig oder nacheinander geplant werden sollen, betrachten wir Parameterschätzprobleme bei Mehrfachexperimenten der Form (4.30) bis (4.32).

Das optimale Versuchsplanungsproblem führt auf ein Optimal-Steuerungsproblem der Form

$$\min_{q,u,w,s} \Phi(C) \tag{5.1}$$

so daß für jedes Experiment $l, l = 1, \ldots, N_{ex}$, die folgenden Bedingungen erfüllt sind:

- Zustands- und Kontrollbeschränkungen

$$ {}^l c(t, {}^l x(t), p, {}^l q, {}^l u(t), {}^l w) \geq 0, \quad t \in [{}^l t_0; {}^l t_{f_l}], \tag{5.2}$$

- Beschränkungen an das zugrundeliegende Parameterschätzproblem

$$ {}^{l}d(p, {}^{l}s, {}^{l}q, {}^{l}u) = 0, \tag{5.3} $$

- und 0-1 oder gemischt-ganzzahlige Bedingungen

$$ {}^{l}w \in \{0,1\}^{n_{m_l}} \text{ oder } {}^{l}w \in \{0,1,2,\ldots, {}^{l}N_w\}^{n_{m_l}}. \tag{5.4} $$

Die Lösungstrajektorie ${}^{l}x(t)$ des DAE-Systems wurde bereits für das Parameterschätzproblem durch die endliche Anzahl von Variablen ${}^{l}s$ parametrisiert. In dem Vektor der Nebenbedingungen $c = ({}^{1}c^T, \ldots, {}^{N_{ex}}c^T)^T$ fassen wir die Beschränkungen an die Zustandsvariablen, an die Steuergrößen und Steuerfunktionen, Kostenbeschränkungen und eventuell auftretende Innere-Punkt-Bedingungen zusammen. Für die Auswertung von Zustandsbeschränkungen in c muß die Lösung des DAE-Systems bereitgestellt werden, für die Auswertung der Zielfunktion zusätzlich auch die Ableitungen der Lösungstrajektorie nach den Parametern und den Parametrisierungsvariablen s.

Im Parameterschätzproblem ist die Standardabweichung ${}^{l}\sigma_{ij}$ des Meßfehlers zu einer gegebenen Messung ein fester Wert. Im Versuchsplanungsproblem hingegen kann die Standardabweichung von den Zustandsvariablen und den Steuergrößen abhängen und wird somit als Funktion

$$ \begin{aligned} {}^{l}\varsigma_{ij} : I\!R^{n_{y_i}+n_{z_i}} \times I\!R^{n_{q_i}} &\to I\!R^{n_{m_{ij}}}, \\ (x_i, q_i) &\mapsto {}^{l}\varsigma_{ij}(x_i, q_i) \end{aligned} $$

betrachtet. Hiermit können zum Beispiel auch Messungen mit einem relativen Meßfehler formuliert werden.

5.1.1 Zielfunktional

Zur Beurteilung der Güte von Versuchsplänen benutzte Kiefer [Kie75] die Formulierung des Matrixmittels.

Definition 5.1.1 (Matrixmittel)
Das Matrixmittel *einer symmetrischen und positiv semi-definiten Matrix M mit Eigenwerten $\lambda_1, \ldots, \lambda_{n_M}$ ist definiert als*

$$ \phi_k(M) = \left(\frac{1}{n_M} \sum_{i=1}^{n_M} \lambda_i^k \right)^{1/k}, \; 0 < k \leq \infty. $$

Falls die Matrix M symmetrisch und positiv definit ist, so ist auch das 0-te Matrixmittel definiert mit

$$ \phi_0(M) = \lim_{k \to 0} \left(\frac{1}{n_M} \sum_{i=1}^{n_M} \lambda_i^k \right)^{1/k} = (\det\, M)^{1/n_M}. $$

Für die Optimierung wählen wir die aus der statistischen Versuchsplanung bekannten Gütekriterien, die dem Matrixmittel für ein $k \in [0, \infty]$ entsprechen:

- **A-Kriterium:** Minimiere die durchschnittliche Varianz der zu schätzenden Parameter
$$\Phi_A(C) = \phi_1(C) = \frac{1}{l_1} \text{ Spur } (C).$$
Das A-Kriterium minimiert die durchschnittliche Halbachse des Konfidenzellipsoids. Das Gütekriterium hat den Vorteil, daß nur die Diagonalelemente der Kovarianzmatrix berechnet werden müssen. Für die in der Optimierung benötigten Ableitungen müssen allerdings trotzdem alle Elemente der Kovarianzmatrix ausgewertet werden.

- **E-Kriterium:** Minimiere den größten Eigenwert der Kovarianzmatrix
$$\Phi_E(C) = \phi_\infty(C) = \max_{i=1}^{l_1} \lambda_i(C).$$
Das E-Kriterium minimiert die größte Ausdehnung des Konfidenzellipsoids.

- **D-Kriterium:** Minimiere die Determinante der Kovarianzmatrix. Im Falle unbeschränkter Parameterschätzprobleme definieren wir
$$\Phi_D(C) = \phi_0(C) = (\det\, C)^{\frac{1}{l_1}}.$$
Für beschränkte Parameterschätzprobleme ist die Kovarianzmatrix singulär und die Determinante hiervon immer gleich Null. In diesem Fall beschränken wir uns auf einen Unterraum $K^T \cdot (p, s)$ der zu schätzenden Größen. Die Matrix $K \in R^{l_1 \times l_K}$ habe dabei Vollrang. Die zugehörige restringierte Kovarianzmatrix hat die Gestalt $C_K = Cov\left(K^T \cdot (p, s)\right) = K^T\, Cov(p, s)\, K$. Für reguläre $C_K \in I\!R^{l_K \times l_K}$ definieren wir das D-Kriterium analog
$$\Phi_D(C_K) = (\det\, C_K)^{\frac{1}{l_K}}.$$
Das D-Kriterium minimiert das Volumen des Konfidenzellipsoids, das durch die Kovarianzmatrix C beziehungsweise C_K beschrieben wird. Ein Nachteil ist dabei, daß das Konfidenzellipsoid zum optimierten Versuchsplan lang und schmal sein kann.

In vielen Fällen ist der Benutzer an den Standardabweichungen beziehungsweise den sogenannten Konfidenzintervallen (siehe Abschnitt 4.6) der zu schätzenden Größen interessiert. Hierzu wurde von Lohmann et al. [LBS92] ein zusätzliches Kriterium eingeführt:

- **Min-max-Kriterium:** Minimiere die Wurzel des größten Diagonalelements der Kovarianzmatrix
$$\Phi_M(C) = \max_{i=1}^{l_1} \sqrt{C_{ii}}.$$
Die Wurzeln der Diagonalelemente der Kovarianzmatrix $\theta_i = \sqrt{C_{ii}}$ geben eine Näherung der Standardabweichungen der zu schätzenden Größen an. Das durch

die genäherte Kovarianzmatrix C beschriebene Konfidenzellipsoid wird von einem Quader mit Seitenlängen $2 \cdot \theta_i$ eingeschlossen.

Das Min-max-Kriterium minimiert die maximale Standardabweichung der zu schätzenden Größen.

Ein weiteres Kriterium könnte zum Beispiel die Minimierung einer Auswahl der Elemente der Korrelationsmatrix sein.

Bemerkung 5.1.1 (Skalierung der Parameter)
Sowohl beim A-, E-, als auch beim Min-max-Kriterium muß darauf geachtet werden, daß alle Parameterwerte etwa von der gleichen Größenordnung sind. Der Vorteil hierbei ist, daß der Benutzer selbst über die Skalierung der Parameter Einfluß auf die von ihm gewünschten, eventuell unterschiedlich großen Standardabweichungen nehmen kann. Sollen einige der Parameter mit größerer Signifikanz geschätzt werden als die restlichen, so sollten diese einen größeren Wert gegenüber den restlichen Parameterwerten haben. Die Experimente werden dann so geplant, daß gerade diese Parameter möglichst genau bestimmt werden können.

Im Gegensatz hierzu ist das D-Kriterium skalierungsinvariant: Seien die Parameter $\theta = K^T \cdot (p, s)$ *(*$K = I$ *im unbeschränkten Fall beziehungsweise* K *wie oben definiert bei beschränkten Parameterschätzproblemen) reparametrisiert durch* $H^T \cdot \theta$, $H \in \mathbb{R}^{n_\theta \times n_\theta}$, *dann gilt für die Kovarianzmatrix* $C(H^T \cdot \theta) = H^T C H$ *und somit* $\det C(H^T \cdot \theta) = \det C(\theta) (\det H)^2$. *Für konstante Matrizen* H *kann die Form eines langen und schmalen Konfidenzellipsoids nicht durch eine lineare Umparametrisierung verändert werden.*

5.1.2 Optimierungsvariablen

Freie Variablen des Optimierungsproblems sind die folgenden das Experiment bestimmenden Größen und die Auswahl der Messungen.

- **Steuergrößen** q**:** Zeitlich konstante Einflußgrößen im Experiment. Dies sind zum Beispiel die Anfangszusammensetzung im Reaktor oder in den Zulaufgefäßen oder die Anfangstemperatur beziehungsweise die Temperatur im Reaktor bei isothermer Fahrweise.

- **Steuerfunktionen** $u(t)$**:** Das Experiment kann häufig auch durch zeitlich variable verfahrenstechnische Komponenten beeinflußt werden, wie etwa der Volumenstrom von Zuläufen oder Abläufen oder das Temperaturprofil einer Heizung beziehungsweise Kühlung.

- **Gewichte** w**:** Die Gewichte beschreiben eine Auswahl von Messungen aus dem vorgegebenen Satz an möglichen Messungen und Meßzeitpunkten. Die Gewichte sind in der Regel 0-1-Variablen, aber auch allgemeine ganzzahlige Variablen sind möglich, wenn eine Messung zum selben Zeitpunkt mehrmals durchgeführt werden kann.

- **Parametrisierungsvariablen s:** Zur Parametrisierung der unendlich-dimensionalen Lösung des DAE-Systems werden die Variablen s, wie in Abschnitt 4.1.1 zur numerischen Lösung des Parameterschätzproblems beschrieben, eingeführt. Im Falle einer Mehrzieldiskretisierung werden an jedem Mehrzielknoten τ_{ij} Variablen s_{ij} für die Anfangswerte $x(\tau_{ij})$ eingeführt, im Falle eines Single-Shooting-Ansatzes wird die Lösung des DAE-Systems nur über die Anfangswerte $s_0 = x(t_0)$ parametrisiert. Die Schätzungen für die zusätzlich eingeführten Variablen s sind wiederum Zufallsvariablen und in erster Näherung normalverteilt wie auch die Schätzungen für die unbekannten Parameter p. Mit Hilfe der Mehrzieldiskretisierung läßt sich nicht nur die statistische Güte der zu schätzenden Parameter maximieren, sondern auch von Komponenten der Lösungstrajektorie zu bestimmten Zeitpunkten τ_{ij}. Dies ist zum Beispiel für die Vorhersage von Komponenten sinnvoll, die nicht direkt gemessen werden können oder um die Zuverlässigkeit des Ergebnisses bei einer anschließenden Optimierung (z. B. Maximierung der Ausbeute oder Minimierung von Abfallprodukten) zu erhöhen. Allerdings wächst der Aufwand zur Berechnung der Kovarianzmatrix und insbesondere zur Berechnung ihrer Ableitung mit der Anzahl der Variablen s. Deshalb wird für die Versuchsplanung eine Parametrisierung derart vorgenommen, daß nur die interessierenden Größen mit in die Kovarianzmatrix aufgenommen werden. Formal werden die Variablen insbesondere für alle Anfangswerte eingeführt, jedoch wird für die Aufstellung der Jacobi-Matrix J (und somit auch der Kovarianzmatrix C) ein reduzierter Ansatz verwendet, der insbesondere die zu den festen Anfangswerten gehörigen Variablen eliminiert. Die Jacobi-Matrix ist dann von der Dimension der Anzahl der Messungen n_{mess} und der interessierenden Variablen n_v, die Kovarianzmatrix von der Dimension $n_v \times n_v$.

5.1.3 Nebenbedingungen

Die Optimierungsvariablen müssen meist zusätzliche Nebenbedingungen erfüllen, sei es durch Grenzen an den Prozeß im Reaktor oder etwa weil das Modell nur innerhalb eines bestimmten Bereichs Gültigkeit besitzt. Die Nebenbedingungen c können Beschränkungen an die Steuergrößen, an die Steuerfunktionen, an die Gewichte und an die Lösungstrajektorie des DAE-Systems enthalten.

Beschränkungen an die Steuergrößen sind etwa Grenzen an die Anfangskonzentrationen oder an Funktionen der verschiedenen Steuergrößen untereinander, etwa die Molverhältnisse von zwei Spezies.

Beschränkungen an die Steuerfunktionen können zum Beispiel die minimale/maximale Zulaufmenge oder Zulaufrate sein oder die minimale/maximale Temperatur einer Heiz- bzw. Kühlschlange.

Da die Messungen sehr teuer und zeitaufwendig sind, wird der Versuchsplan aus mehreren möglichen Messungen und Meßzeitpunkten einige wenige auswählen. Dabei können die Gewichte pro Beobachtungsfunktion, pro Experiment und/oder insgesamt beschränkt

werden. Auch die etwas komplexere Formulierung als Beschränkung an Kosten etwa von Edukten und von Messungen kann hiermit formuliert werden.

Zusätzlich können Beschränkungen an die Zustandsvariablen $x(t)$ und somit an die Lösungstrajektorie des DAE-Systems auftreten, etwa um zu verhindern, daß das Modell seinen Gültigkeitsbereich verläßt oder Sicherheitsbestimmungen verletzt werden. Die Beschränkungen an die Steuerfunktionen und die Zustandsvariablen sollen zunächst über den gesamten Beobachtungszeitraum eingehalten werden. Für die numerische Lösung werden die Zustandsbeschränkungen durch Innere-Punkt-Bedingungen an diskreten Punkten ersetzt, die Steuerfunktionen werden geeignet parametrisiert und die Beschränkungen an die Parametrisierungsvariablen formuliert.

5.2 Direkter Ansatz

Das Versuchsplanungsproblem (5.1) bis (5.4) ist ein unendlich-dimensionales beschränktes Optimal-Steuerungsproblem in den Variablen $q, u(t), w$ und s. Zur Lösung hierfür wählen wir einen direkten Ansatz. Wir approximieren die Steuerfunktionen durch stückweise stetige Funktionen mit einer endlichen Anzahl von Variablen und ersetzen die Zustandsbeschränkungen und Beschränkungen an die Steuerfunktionen durch Innere-Punkt-Bedingungen an diskreten Punkten. Die ganzzahligen Bedingungen werden durch eine relaxierte Formulierung ersetzt.

Die unendlich-dimensionale Lösung des DAE-Systems wurde bereits bei der Anwendung des direkten Ansatzes für die Parameterschätzung durch die Parametrisierung mit Hilfe der Variablen s in eine Funktion überführt, die nur noch von einer endlichen Anzahl von Variablen abhängt. Dabei wird wiederum die relaxierte Formulierung der algebraischen Gleichungen (4.4b) zur Lösung der DAE-Systeme gewählt, um konsistente Anfangswerte in jedem Iterationsschritt der Optimierung zu garantieren. Die Konsistenzbedingungen (4.8) (beziehungsweise (4.33) im Mehrfachexperimentfall) werden als zusätzliche Nebenbedingungen in das resultierede endlich-dimensionale Optimierungsproblem aufgenommen, um eine konsistente Lösung der ursprünglichen DAE-Systeme im Lösungspunkt der Optimierung zu garantieren.

5.2.1 Approximation der Steuerfunktionen

Zur Überführung der Steuerfunktionen in eine endlich-dimensionale Anzahl von Variablen diskretisieren wir den Beobachtungszeitraum $[{}^l t_0, {}^l t_{f_l}]$ wiederum in Teilintervalle $[{}^l\tau_i, {}^l\tau_{i+1}]$, $i = 0, \ldots, m_l - 1$, mit

$$ {}^l t_0 = {}^l\tilde{\tau}_0 < \cdots < {}^l\tilde{\tau}_{m_l} = {}^l t_{f_l}. $$

Auf jedem Teilintervall approximieren wir $u(t)$, $t \in [{}^l\tilde{\tau}_{i_l}, {}^l\tilde{\tau}_{i_l+1}]$, durch eine stetige Funktion mit endlich vielen Freiheitsgraden, zum Beispiel durch ein Polynom $p(t)$. In der

Regel werden hierfür Approximationen durch stückweise konstante oder stückweise lineare Funktionen gewählt. Aber auch andere Approximationen der Steuerfunktionen, die nur von einer endlichen Anzahl von Variablen abhängen, sind möglich. Eventuelle Stetigkeitsbedingungen an den inneren Gitterpunkten ${}^l\tau_{i_l}$ werden mit in den Vektor der Nebenbedingungen aufgenommen.

Im folgenden seien die aus der Approximation der Steuerfunktionen zusätzlich auftretenden Variablen mit in den Vektor der Steuergrößen q aufgenommen.

5.2.2 Diskretisierung der Zustandsbeschränkungen

Beschränkungen an die Lösung des DAE-Systems werden durch Innere-Punkt-Bedingungen zu bestimmten diskreten Zeitpunkten ersetzt. Hierfür definieren wir ein weiteres Gitter

$$ {}^l t_0 = {}^l\bar{\tau}_0 < \cdots < {}^l\bar{\tau}_{m_l} = {}^l t_{f_l}. $$

Dieses muß nicht notwendigerweise das gleiche sein wie das Gitter für die Approximation der Steuerfunktionen oder das Gitter der Mehrzieldiskretisierung.

5.2.3 Relaxierung der ganzzahligen Bedingungen

Zur numerischen Lösung des Optimierungsproblems verwenden wir eine relaxierte Formulierung der ganzzahligen Bedingungen, das heißt

$$ {}^l w \in [0,1]^{n_{m_l}} \quad l = 1, \ldots, N_{ex}, $$

oder

$$ {}^l w_i \in [0, {}^l N_{w_i}] \quad l = 1, \ldots, N_{ex},\ i = 1, \ldots, n_{m_l}. $$

Eine Messung mit doppelter Varianz kann wie eine Messung mit Gewicht 1/2 interpretiert werden. Um im Anschluß an die Optimierung wieder eine ganzzahlige Lösung zu erhalten, wenden wir Rundungs-Heuristiken an. Eine Möglichkeit hierfür ist, die Gewichte pro Meßfunktion mit aufsteigenden Meßzeitpunkten aufzusummieren. Wenn das Gewicht größer als 1 ist (oder eventuell schon größer als 1/2, um eine Verschiebung der Gewichte in Richtung höherer Meßzeitpunkte zu verhindern), setzen wir das aktuelle Gewicht auf 1 und die vorherigen auf 0, verringern den Wert der Summe um 1 und machen weiter im Algorithmus. Eine weitere Möglichkeit wäre, die Gewichte mit den höheren Werten aufzurunden und diejenigen mit niedrigen Gewichten abzurunden unter der Bedingung, daß die Beschränkungen an die Gewichte nicht verletzt werden.

Die Ergebnisse für die betrachteten Beispiele haben allerdings ergeben, daß die Gewichte im optimierten Versuchsplan nahezu ganzzahlig waren und eine Rundung mit den oben beschriebenen Heuristiken leicht möglich war und nur wenig Einfluß auf das Gütekriterium hatte.

5.3 Behandlung des Optimierungsproblems

Das resultierende nichtlineare endlich-dimensional Optimierungsproblem schreiben wir in verkürzter Form als

$$\begin{aligned} &\min_v F_1(v) \\ \text{so daß } &F_2(v) = 0 \\ &F_3(v) \geq 0\,. \end{aligned} \tag{5.5}$$

In dem Vektor $v \in \mathbb{R}^{n_v}$ fassen wir die Vektoren q, die aus den ursprünglich zeitlich konstanten Steuergrößen und die aus der Approximation der Steuerfunktionen zusätzlich eingeführten zeitlich konstanten Größen bestehen, zusammen sowie den Vektor der Variablen s aus der Parametrisierung der DAE-Systeme und die Gewichte w an die Messungen.

$F_1 : \mathbb{R}^{n_v} \to \mathbb{R}$ beschreibt die Zielfunktion. Die Nebenbedingungen des Optimierungsproblems ordnen wir nach Gleichungs- und Ungleichungsbedingungen. Die Funktion $F_2 : \mathbb{R}^{n_v} \to \mathbb{R}^{n_2}$ beinhaltet die Gleichungsnebenbedingungen aus c, die Nebenbedingungen an das zugrundeliegende Parameterschätzproblem – in denen auch die Konsistenzbedingungen für die algebraischen Variablen enthalten sind – und gleichungsbeschränkte Nebenbedingungen an die Steuergrößen wie etwa Stetigkeitsbedingungen der Approximationen der Steuerfunktionen an inneren Gitterpunkten. $F_3 : \mathbb{R}^{n_v} \to \mathbb{R}^{n_3}$ umfaßt die Ungleichungsbedingungen aus c, Innere-Punkt-Bedingungen an die Zustandsvariablen, Schranken an die Meßgewichte und/oder Schranken an die Kosten.

5.3.1 Optimalitätsbedingungen

Seien die Funktionen F_i, $i = 1, \ldots, 3$, mindestens einmal stetig differenzierbar, für Bedingungen zweiter Ordnung seien zusätzlich die zweiten partiellen Ableitungen stetig. Insbesondere seien die Voraussetzungen aus Satz 4.2.2 für das zugrundeliegende Parameterschätzproblem erfüllt. Ähnlich zum Least-Squares-Problem (4.17) definieren wir nachfolgend Optimalitätsbedingungen für das Optimierungsproblem (5.5).

Definition 5.3.1 (Zulässiger Bereich)
Der zulässige Bereich *des Optimierungsproblems (5.5) ist die Menge aller Punkte v, die die Nebenbedingungen erfüllen*

$$\mathcal{M}_{zul} = \{v \in \mathbb{R}^{n_v} | F_2(v) = 0,\, F_3(v) \geq 0\}.$$

Definition 5.3.2 (Active-Set)
Die Menge der aktiven Ungleichungen *in einem Punkt $v \in \mathcal{M}_{zul}$ sind diejenigen Komponenten von F_3, die mit Gleichheit erfüllt sind. Hierfür definieren wir die Indexmenge des Active-Set*

$$\mathcal{I}(v) = \{i \in \{1, \ldots, n_3\} | F_{3,i}(v) = 0\}.$$

Definition 5.3.3 (Lokales Minimum)
Ein Punkt $v^* \in \mathcal{M}_{zul}$ heißt lokales Minimum, *falls eine Umgebung $\mathcal{U}$ von v^* existiert, so daß*

$$F_1(v^*) \leq F_1(v) \quad \forall v \in \mathcal{U} \cap \mathcal{M}_{zul}.$$

Falls die Ungleichung strikt ist, so heißt v^* striktes lokales Minimum.

Definition 5.3.4 (Regulärer Punkt)
Ein Punkt $v^* \in \mathcal{M}_{zul}$ heißt regulär, *falls die Vektoren $\nabla F_{2,i}(v^*), i = 1, \ldots, n_2$, und $\nabla F_{3,i}(v^*), i \in \mathcal{I}(v^*)$, linear unabhängig sind. Diese Bedingung heißt im allgemeinen auch* first-order constraint qualification.

Für das Optimierungsproblem (5.5) ist die *Lagrange-Funktion* von der Form

$$\mathcal{L}(v, \lambda, \mu) = F_1(v) - \lambda^T F_2(v) - \mu^T F_3(v) \tag{5.6}$$

mit *Lagrange-Multiplikatoren* λ und μ.

Satz 5.3.1 (Notwendige Bedingungen)
Sei $v^* \in \mathcal{M}_{zul}$ regulär und lokales Minimum. Dann existieren Vektoren $\lambda \in \mathbb{R}^{n_2}$ und $\mu \in \mathbb{R}^{n_3}$ mit

i) (Notwendige Bedingung 1. Ordnung)

$$\nabla_v \mathcal{L}(v^*, \lambda, \mu) = 0$$

ii) (Komplementarität)

$$\mu^T F_3(v^*) = 0$$

iii) (Notwendige Bedingung 2. Ordnung)

$$w^T \nabla_{vv} \mathcal{L}(v^*, \lambda, \mu)\, w \geq 0$$

$$\forall w \in \{\omega \in \mathbb{R}^{n_v} | \nabla F_{2,i}(v^*)\, \omega = 0, i = 1, \ldots, n_2,\ \nabla F_{3,i}(v^*)\, \omega = 0, i \in \mathcal{I}(v^*)\}.$$

5.3.2 Numerische Lösung des Optimierungsproblems

Das Optimierungsproblem wird mit Verfahren der sequentiellen quadratischen Programmierung (SQP) gelöst: Wir starten mit einer Anfangsschätzung v^0 und berechnen eine neue Iterierte aus

$$v^{k+1} = v^k + r^k\, \Delta v^k, \quad k = 0, 1, \ldots. \tag{5.7}$$

Die Schrittweite $r^k \in (0, 1]$ wird mit Hilfe von Liniensuchmethoden gewonnen. Die Suchrichtung Δv^k löst das quadratische Teilproblem (QP)

$$\begin{aligned} \min_{\Delta v} \frac{1}{2} \Delta v^T\, A^k\, \Delta v + \nabla F_1(v^k)^T\, \Delta v \\ \text{so daß } F_2(v^k) + \nabla F_2(v^k)^T\, \Delta v = 0 \\ F_3(v^k) + \nabla F_3(v^k)^T\, \Delta v \geq 0\,. \end{aligned} \tag{5.8}$$

Matrix A^k beschreibt dabei eine Approximation der Hesse-Matrix der Lagrange-Funktion (5.6).

Zur numerischen Lösung des Optimierungsproblems (5.5) wurden zum einen der Algorithmus E04UCF aus der NAG-Programmbibliothek [NAG91] verwendet, zum anderen das von Gill et al. [GMS97a, GMS97b] entwickelte SQP-Verfahren SNOPT. Beide Programmpakete bieten zwar die Möglichkeit, die Gradienten ∇F_i, $i = 1, \ldots, 3$, mit Hilfe numerischer Differenzen zu berechnen, jedoch bedeutet dieser Black-Box-Ansatz einen hohen Rechenaufwand bei gleichzeitig sehr ungenauen Ableitungen – oft nicht einmal eine Stelle Genauigkeit.

Im folgenden wird eine effiziente Berechnung der Zielfunktion F_1 und der Nebenbedingungen F_2 und F_3 als auch deren Ableitungen nach den Optimierungsvariablen beschrieben, die auf einem geschickten Zusammenspiel von semi-analytischen Ableitungen, Automatischer Differentiation und Interner Numerischer Differentiation beruht. Die Approximation der Hesse-Matrix A^k wird über Update-Verfahren basierend auf den Gradienten der Lagrange-Funktion gewonnen.

5.4 Herleitung der Gradienten von Zielfunktional und Nebenbedingungen

Zur Berechnung der Ableitungen der Nebenbedingungen wenden wir die Kettenregel an

$$\frac{\mathrm{d}F_i}{\mathrm{d}v} = \frac{\partial F_i}{\partial x}\frac{\partial x}{\partial v} + \frac{\partial F_i}{\partial v},\ i = 2, 3.$$

Die partiellen Ableitungen der Nebenbedingungen $F_i, i = 2, 3$, nach den Variablen x und v werden mit Automatischer Differentiation bereitgestellt. Die Berechnung der Ableitungen der Lösungstrajektorie $x(t)$ nach den Optimierungsvariablen v wird in Kapitel 6 näher beschrieben. Die Trajektorie $x(t)$ hängt nicht von den Meßgewichten w ab, somit sind die diesbezüglichen Ableitungen gleich Null und brauchen natürlich nicht berechnet zu werden.

Zur Herleitung der Ableitungen des Zielfunktionals nach den Optimierungsvariablen v wenden wir zunächst wiederum die Kettenregel an

$$\frac{\mathrm{d}F_1}{\mathrm{d}v} = \frac{\partial \Phi}{\partial C}\frac{\partial C}{\partial J}\frac{\mathrm{d}J}{\mathrm{d}v}\,. \tag{5.9}$$

Auf die Berechnung der einzelnen Terme mit Hilfe von semi-analytischen Ableitungen, Automatischer Differentiation und Interner Numerischer Differentiation wird in den folgenden Abschnitten näher eingegangen.

5.4.1 Ableitung des Gütekriteriums nach der Kovarianzmatrix

Für die Berechnung der Ableitung des Gütekriteriums Φ nach der Kovarianzmatrix C stehen explizite Formeln zur Verfügung:

- *A-Kriterium:*
$$\frac{\mathrm{d}\Phi_A(C)}{\mathrm{d}v} = \frac{1}{n_v}\,\mathrm{Spur}\left(\frac{\mathrm{d}C}{\mathrm{d}v}\right).$$

- *E-Kriterium:* Sei $\lambda_{max}(C)$ ein einfacher Eigenwert von C. Dann ist die Ableitung von Φ_E gegeben durch
$$\frac{\mathrm{d}\Phi_E(C)}{\mathrm{d}v} = z^T\left(\frac{\mathrm{d}C}{\mathrm{d}v}\right)z,$$
wobei z den normierten Eigenvektor zum Eigenwert $\lambda_{max}(C)$ beschreibt.

- *D-Kriterium:*
$$\frac{\mathrm{d}\Phi_D(C_K)}{\mathrm{d}v} = \frac{1}{l_K}\left(\det C_K\right)^{\frac{1}{l_K}}\sum_{i=1}^{l_K}\sum_{j=1}^{l_K}\left(C_K^{-1}\right)_{ij}\cdot\left(\frac{\mathrm{d}C_K}{\mathrm{d}v}\right)_{ij}.$$

- *Min-max-Kriterium:* Das Optimierungsproblem für das Min-max-Kriterium
$$\begin{aligned}\min\quad & \max_{i=1}^{l_1}\sqrt{C_{ii}}\\ & + \text{Nebenbedingungen}\end{aligned}$$
ist nicht differenzierbar. Es läßt sich aber überführen in das Optimierungsproblem
$$\begin{aligned} min\quad & \delta && \text{(5.10a)}\\ \text{so daß}\quad & \sqrt{C_{ii}} \le \delta,\ i = 1,\ldots,l_1, && \text{(5.10b)}\\ & + \text{Nebenbedingungen.} && \text{(5.10c)}\end{aligned}$$
Die Ableitung der zusätzlichen Nebenbedingungen (5.10b) ist dann
$$\frac{\mathrm{d}\sqrt{C_{ii}}}{\mathrm{d}v} = \frac{1}{2\sqrt{C_{ii}}}\,\frac{\mathrm{d}C_{ii}}{\mathrm{d}v}.$$

Beweis: Für das A-, D- und E-Kriterium siehe zum Beispiel Pázman [Páz86] und Lohmann [Loh93], für die Ableitung des Min-max-Kriteriums siehe Lohmann [Loh93].

5.4.2 Ableitung der Kovarianzmatrix nach der Jacobi-Matrix

Im folgenden leiten wir Formeln für die Berechnung des zweiten Terms in (5.9) her. Mit Hilfe von Zerlegungen der Jacobi-Matrix J des zugrundeliegenden Parameterschätzproblems können sowohl die Kovarianzmatrix C als auch deren Ableitungen nach der Jacobi-Matrix J geschickt berechnet werden. Die Kovarianzmatrix
$$C \;=\; J^{+}\begin{pmatrix} I & 0\\ 0 & 0\end{pmatrix}J^{+T}$$

läßt sich darstellen als

$$\begin{aligned} C &= \begin{pmatrix} I & 0 \end{pmatrix} \begin{pmatrix} J_1^T J_1 & J_2^T \\ J_2 & 0 \end{pmatrix}^{-1} \begin{pmatrix} J_1^T J_1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} J_1^T J_1 & J_2^T \\ J_2 & 0 \end{pmatrix}^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \\ &=: I_1 \cdot M^{-1} \cdot S \cdot M^{-1} \cdot I_1^T . \end{aligned} \tag{5.11}$$

Die Ableitung der Kovarianzmatrix C nach den Optimierungsvariablen v ist somit (mit $\frac{\mathrm{d}M^{-1}}{\mathrm{d}v} = -M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} M^{-1}$)

$$\frac{\mathrm{d}C}{\mathrm{d}v} = -I_1 M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} M^{-1} S M^{-1} I_1^T + I_1 M^{-1} \frac{\mathrm{d}S}{\mathrm{d}v} M^{-1} I_1^T - I_1 M^{-1} S M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} M^{-1} I_1^T \tag{5.12}$$

mit

$$\frac{\mathrm{d}M}{\mathrm{d}v} = \begin{pmatrix} \frac{\mathrm{d}}{\mathrm{d}v}\left(J_1^T J_1\right) & \frac{\mathrm{d}}{\mathrm{d}v} J_2^T \\ \frac{\mathrm{d}}{\mathrm{d}v} J_2 & 0 \end{pmatrix} \text{ und } \frac{\mathrm{d}S}{\mathrm{d}v} = \begin{pmatrix} \frac{\mathrm{d}}{\mathrm{d}v}(J_1^T J_1) & 0 \\ 0 & 0 \end{pmatrix}.$$

Umformung von (5.11) ergibt

$$C \;=\; (I_1 M^{-1}) \left((I_1 M^{-1}) S\right)^T.$$

Die Ableitung der Kovarianzmatrix läßt sich weiter umformen zu

$$\begin{aligned} \frac{\mathrm{d}C}{\mathrm{d}v} &= (I_1 M^{-1}) \left(-\frac{\mathrm{d}M}{\mathrm{d}v} M^{-1} S + \frac{\mathrm{d}S}{\mathrm{d}v} - S M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} \right) (I_1 M^{-1})^T \\ &= (I_1 M^{-1}) \left[(I_1 M^{-1}) \left(-\frac{\mathrm{d}M}{\mathrm{d}v} M^{-1} S + \frac{\mathrm{d}S}{\mathrm{d}v} - S M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} \right)^T \right]^T \\ &= (I_1 M^{-1}) \left[(I_1 M^{-1}) \left(-\left(S M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} \right)^T + \frac{\mathrm{d}S}{\mathrm{d}v} - S M^{-1} \frac{\mathrm{d}M}{\mathrm{d}v} \right)^T \right]^T \\ &= (I_1 M^{-1}) \left[(I_1 M^{-1}) \left(-\left(S_1 (I_1 M^{-1}) \frac{\mathrm{d}M}{\mathrm{d}v} \right)^T + \frac{\mathrm{d}S}{\mathrm{d}v} - S_1 (I_1 M^{-1}) \frac{\mathrm{d}M}{\mathrm{d}v} \right)^T \right]^T \end{aligned} \tag{5.13}$$

mit $S = (S_1 | S_2) = \left(\begin{array}{c|c} J_1^T J_1 & 0 \\ 0 & 0 \end{array} \right)$.

Für die Berechnung der Kovarianzmatrix und ihrer Ableitungen wird die Matrix $\widetilde{M} = (I_1 M^{-1})$ mehrere Male von rechts mit unterschiedlichen Matrizen multipliziert.

Sei nun $\Lambda = (\Lambda_1^T, \Lambda_2^T)^T$ die von rechts multiplizierte Matrix. Wir lösen zunächst formal das System

$$\begin{pmatrix} J_1^T J_1 & J_2^T \\ J_2 & 0 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \end{pmatrix}. \tag{5.14}$$

Da immer eine Multiplikation der Form

$$\widetilde{M} \Lambda = \begin{pmatrix} I & 0 \end{pmatrix} M^{-1} \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \end{pmatrix} = \begin{pmatrix} I & 0 \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = X_1$$

vorgenommen wird, benötigen wir nur die Berechnung von X_1.

Aufgrund der Voraussetzungen von Satz 4.2.2 ist die Matrix $J_1^T J_1$ positiv definit und Matrix J_2 hat Vollrang. Wir nehmen zunächst eine QR-Zerlegung von J_2^T vor und zerlegen anschließend das transformierte System in der ersten Zeile von (5.14). Eine ähnliche Vorgehensweise wurde bereits von Raum [Rau97] dargestellt, siehe auch Bauer et al. [BKBS98, BBKS99b].

Aus der zweiten Zeile von (5.14) folgt $J_2 X_1 = \Lambda_2$. Wendet man auf J_2^T eine QR-Zerlegung an ($J_2^T = Q^T \begin{pmatrix} R^T \\ 0 \end{pmatrix}$), so erhält man

$$\begin{pmatrix} R & 0 \end{pmatrix} Q\, X_1 = \Lambda_2.$$

Mit

$$Q\, X_1 = Y = \begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix}$$

folgt $\Lambda_2 = \begin{pmatrix} R & 0 \end{pmatrix} Y = R\, Y_1$ und somit

$$Y_1 = R^{-1}\, \Lambda_2. \tag{5.15}$$

Aus der ersten Zeile von (5.14) folgt nun

$$\begin{aligned} J_1^T J_1\, Q^T\, Y + J_2^T X_2 &= \Lambda_1 \\ \Longrightarrow\; J_1^T J_1\, Q_1^T\, Y_1 + J_1^T J_1\, Q_2^T\, Y_2 + J_2^T X_2 &= \Lambda_1. \end{aligned} \tag{5.16}$$

Wir multiplizieren (5.16) von links mit Q_2 und erhalten

$$Q_2\, J_1^T J_1\, Q_1^T\, Y_1 + Q_2\, J_1^T J_1\, Q_2^T\, Y_2 + \underbrace{Q_2\, J_2^T}_{=0} X_2 = Q_2\, \Lambda_1.$$

Die Zerlegung von J_2 wurde genau so gewählt, daß $Q_2\, J_2^T = 0$, und somit gilt

$$Q_2\, J_1^T J_1\, Q_2^T\, Y_2 = Q_2\, \Lambda_1 - Q_2\, J_1^T J_1\, Q_1^T\, Y_1. \tag{5.17}$$

Durch Zerlegung von $Q_2\, J_1^T J_1\, Q_2^T$ (mit LU-, QR- oder Cholesky-Zerlegung) erhalten wir aus Gleichungssystem (5.17) die Matrix Y_2. Anschließend berechnen wir

$$X_1 = Q^T\, Y = Q_1^T\, Y_1 + Q_2^T\, Y_2. \tag{5.18}$$

5.4.3 Ableitung der Jacobi-Matrix nach den Optimierungsvariablen

In Formel (5.13) treten noch die Ableitungen von M und S, beziehungsweise die Ableitungen der Jacobi-Matrizen J_1 und J_2 nach den Optimierungsvariablen v auf.

Wir schreiben die Jacobi-Matrix J_1 des Least-Squares-Funktionals (4.30) verkürzt als

$$J_1 = \frac{\mathrm{d}r}{\mathrm{d}(p,s)} = -W\, \Sigma^{-1} \left(\frac{\partial b}{\partial x}\, \frac{\partial x}{\partial (p,s)} + \frac{\partial b}{\partial (p,s)} \right), \tag{5.19}$$

wobei $\Sigma^2 = \operatorname{diag}\left({}^l\varsigma_{ij}({}^l x_i(t_{ij}), {}^l q)^2\right)$ die Kovarianzmatrix der Meßfehler, $b = {}^l b_{ij}(t_{ij}, {}^l x_i(t_{ij}), p, {}^l q)$ den Vektor der Meßfunktionen und $W = \operatorname{diag}(\sqrt{{}^l w_{ij}})$ die Gewichte an die Messungen beschreibt.

Die Ableitung der Jacobi-Matrix J_1 nach der k-ten Komponente der Steuergrößen q ist somit

$$\begin{aligned}\frac{\mathrm{d}J_1}{\mathrm{d}q_k} = W\,\Sigma^{-2}\cdot\left(\operatorname{diag}\left(\frac{\partial\,{}^l\varsigma_{ij}}{\partial x}\,\frac{\partial x}{\partial q_k}+\frac{\partial\,{}^l\varsigma_{ij}}{\partial q_k}\right)\right)\cdot\left(\frac{\partial b}{\partial x}\,\frac{\partial x}{\partial(p,s)}+\frac{\partial b}{\partial(p,s)}\right)\\ -\;W\,\Sigma^{-1}\left(\left(\frac{\partial^2 b}{\partial x^2}\,\frac{\partial x}{\partial q}\right)\frac{\partial x}{\partial(p,s)}+\frac{\partial^2 b}{\partial q\partial x}\,\frac{\partial x}{\partial(p,s)}\right.\\ \left.+\,\frac{\partial b}{\partial x}\,\frac{\partial^2 x}{\partial q\partial(p,s)}+\frac{\partial^2 b}{\partial x\partial(p,s)}\,\frac{\partial x}{\partial q}+\frac{\partial^2 b}{\partial q\partial(p,s)}\right).\end{aligned} \tag{5.20}$$

Für die Ableitung nach den Gewichten w berechnen wir komponentenweise $\left((\mathrm{d}/\mathrm{d}w_k\,(J_1^T\,J_1)\right)$. Dabei sei w_k die k-te Komponente von ${}^l w_{ij}$ und e_k der k-te Einheitsvektor.

$$\frac{\mathrm{d}}{\mathrm{d}w_k}\,(J_1^T\,J_1) = \left(\frac{\partial b}{\partial x}\,\frac{\partial x}{\partial(p,s)}+\frac{\partial b}{\partial(p,s)}\right)^T\Sigma^{-1}\,e_k\,e_k^T\,\Sigma^{-1}\left(\frac{\partial b}{\partial x}\,\frac{\partial x}{\partial(p,s)}+\frac{\partial b}{\partial(p,s)}\right). \tag{5.21}$$

Die ersten und zweiten Ableitungen der Modellfunktionen b in (5.19) bis (5.21) werden dabei direkt über Richtungsableitungen berechnet.

Die Ableitungen der Nebenbedingungen $d = \left({}^1 d^T, \ldots, {}^{N_{ex}} d^T\right)^T$ des Parameterschätzproblems mit ${}^l d = {}^l d({}^l x({}^l t_0), \ldots, {}^l x({}^l t_{f_l}), p, {}^l s, {}^l q))$, $l = 1, \ldots, N_{ex}$, nach den Steuergrößen q führen auf

$$\begin{aligned}\frac{\mathrm{d}J_2}{\mathrm{d}q} = \frac{\mathrm{d}}{\mathrm{d}q}\left(\frac{\mathrm{d}d}{\mathrm{d}(p,s)}\right) = \frac{\mathrm{d}}{\mathrm{d}q}\left(\frac{\partial d}{\partial x}\,\frac{\partial x}{\partial(p,s)}+\frac{\partial d}{\partial(p,s)}\right)\\ =\left(\left(\frac{\partial^2 d}{\partial x^2}\,\frac{\partial x}{\partial q}\right)\frac{\partial x}{\partial(p,s)}+\frac{\partial^2 d}{\partial q\partial x}\,\frac{\partial x}{\partial(p,s)}\right.\\ \left.+\,\frac{\partial d}{\partial x}\,\frac{\partial^2 x}{\partial q\partial(p,s)}+\frac{\partial^2 d}{\partial x\partial(p,s)}\,\frac{\partial x}{\partial q}+\frac{\partial^2 d}{\partial q\partial(p,s)}\right).\end{aligned} \tag{5.22}$$

Von den Meßgrößen w hängt J_2 nicht ab.

Die Berechnung der ersten und zweiten Ableitungen der Lösungstrajektorie $x(t)$ des DAE-Systems nach den Parametern p, den Parametrisierungsvariablen s und den Steuergrößen q kann durch eine geschickte Kombination aus Interner Numerischer Differentiation und Automatischer Differentiation mit der von der Optimierung geforderten Genauigkeit effizient berechnet werden (siehe zum Beispiel Bauer et al. [BBKS99a, BBKS99b]). Unterschiedliche Vorgehensweisen und die Effizienz in Abhängigkeit von den speziellen Anforderungen werden im folgenden Kapitel diskutiert.

5.5 Sequentielles Design

Das Zielfunktional bei der optimalen Versuchsplanung ist eine Funktion auf der genäherten Kovarianzmatrix C aus (4.35). Der Fehler, der aufgrund der Approximation der Kovarianzmatrix auftritt, besteht aus zwei Komponenten:

1. Die Approximation der Kovarianzmatrix beschreibt das Konfidenzellipsoid für das linearisierte Parameterschätzproblem (4.19) und nicht das nichtlineare Konfidenzgebiet. Das mit Hilfe der genäherten Kovarianzmatrix beschriebene Konfidenzellipsoid ist somit eine Approximation erster Ordnung des nichtlinearen Konfidenzgebiets (siehe zum Beispiel Bock [Boc87] und Lohmann [Loh88]). Emig und Hosten [EH74] verglichen das nichtlineare Konfidenzgebiet mit dem linearisierten Konfidenzellipsoid und mit den gemeinsamen Konfidenzintervallen und zeigten für Beispiele, bei denen die Parameter stark nichtlinear in das Modell eingehen, daß das „exakte“ nichtlineare Konfidenzgebiet sehr asymmetrisch um den aktuellen Punkt liegen kann.

2. Die Linearisierung erfolgt im aktuellen Punkt $(\hat{p}, \hat{s})$ und nicht im wahren (unbekannten) Punkt (p^*, s^*).

Da die Modellantwort b in der Regel von den Zustandsvariablen x und somit implizit und nichtlinear von den aktuellen Werten der Parameter $\hat{p}$ und der Parametrisierungsvariablen $\hat{s}$ abhängt, ist der berechnete Versuchsplan zwar optimal für die Linearisierung im aktuellen Punkt $(\hat{p}, \hat{s})$, nicht aber unbedingt für den wahren Wert (p^*, s^*).

Beispiel 5.1 (Lotka-Volterra-Testbeispiel)
Für ein Lotka-Volterra-Modell

$$\begin{aligned} \dot{y}_1 &= f\, y_1 - k_1\, y_1\, y_2 \\ \dot{y}_2 &= k_1\, y_1\, y_2 - k_2\, y_2 \end{aligned}$$

sollen die Parameter k_1 und k_2 bestimmt werden. Die Frequenz f sei als bekannt ($f = 1$) vorausgesetzt.

Wir planen ein Experiment zur Bestimmung beider Parameter. Dabei können die Zustandsvariablen y_1 und y_2 über einen Beobachtungszeitraum von $[0, 5]$ jeweils maximal 20 Mal alle 0.25 Zeiteinheiten gemessen werden. Versuchsplanungsgrößen sind die Anfangswerte $y_1(0)$ und $y_2(0)$ und die Gewichte an die möglichen Messungen. Dabei sollen aus den 40 möglichen Messungen maximal 20 ausgewählt werden.

Wir planen ein Experiment (E-optimal) mit Startschätzungen $k_1 = k_2 = 1$. Das Gütekriterium des optimierten Versuchsplans ist $3.0 \cdot 10^{-4}$. Das Problem ist, daß der Versuchsplan für die Startschätzungen zwar sehr gut ist, falls die wahren Werte für die Parameter jedoch etwas weiter davon entfernt sind, kann aus dem Experiment eventuell nur noch wenig Information erhalten werden. Abbildung 5.1 zeigt den Wert des Gütekriteriums für den

für die Startschätzungen $k_1 = k_2 = 1$ optimierten Versuchsplan in Abhängigkeit von den Parametern k_1 und k_2. Man sieht, daß falls die Werte der „wahren" Parameter weiter weg von $(1,1)^T$ liegen, diese zum Teil nicht mehr aus den Meßdaten geschätzt werden können.

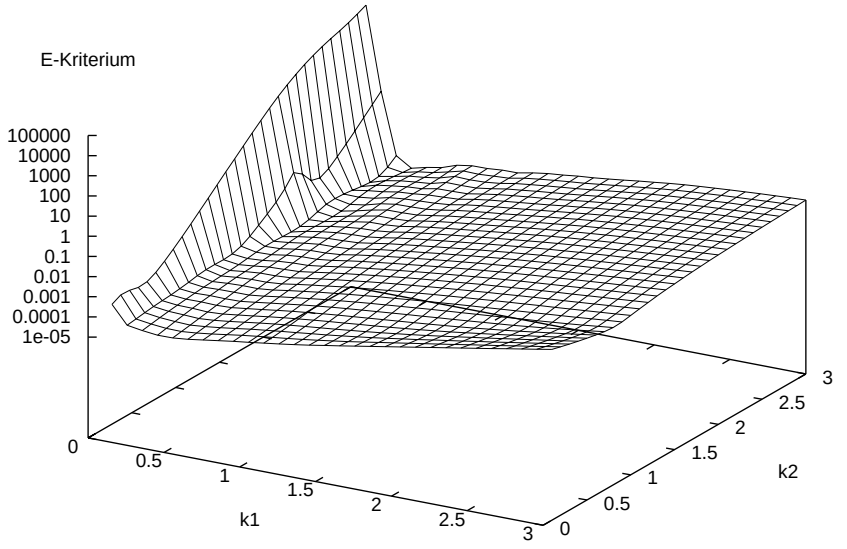


Abbildung 5.1: Werte des Gütekriteriums in Abhängigkeit der Parameter k_1 und k_2 für einen in $(k_1, k_2)^T = (1,1)^T$ optimierten Versuchsplan für das Lotka-Volterra-Testbeispiel

Gehen wir davon aus, daß die „wahren" Parameter im Bereich $[0.5, 3]$ liegen, so sollte ein optimierter Versuchsplan möglichst für den ganzen Bereich noch gute Information für die zu schätzenden Parameter liefern. Wir haben deshalb einen Versuchsplan berechnet, der nicht nur für die Startschätzungen $(k_1, k_2)^T = (1,1)^T$ optimal ist, sondern das optimale Ergebnis liefert für Parameter, die die Werte $k_1, k_2 \in \{0.5, 3\}$ und $k_1 = k_2 = 1$ annehmen können. Abbildung 5.2 zeigt die Werte des Gütekriteriums in Abhängigkeit von den Parametern k_1 und k_2 für den so optimierten Versuchsplan. Der Wert des Gütekriteriums im Punkt $(k_1, k_2)^T = (1,1)^T$ ist $8.3 \cdot 10^{-4}$, also größer als für den nur für die Startschätzungen $(1,1)^T$ optimierten Versuchsplan, das Gütekriterium ist allerdings für Werte im gesamten Bereich $[0.5, 3]$ noch relativ klein.

Schon von Fedorov [Fed72] wurde zur Behebung dieser Problematik eine sequentielle Vorgehensweise vorgeschlagen, weniger aufgrund der Tatsache, daß bei nichtlinearen Problemen die Güte der Versuchspläne stark von den aktuellen Parameterwerten abhängt, als vielmehr weil sich die experimentellen Anforderungen ändern können (neue Meßapparaturen, andere Lösungsmittel) beziehungsweise auch das Hintergrundwissen über den Prozeß

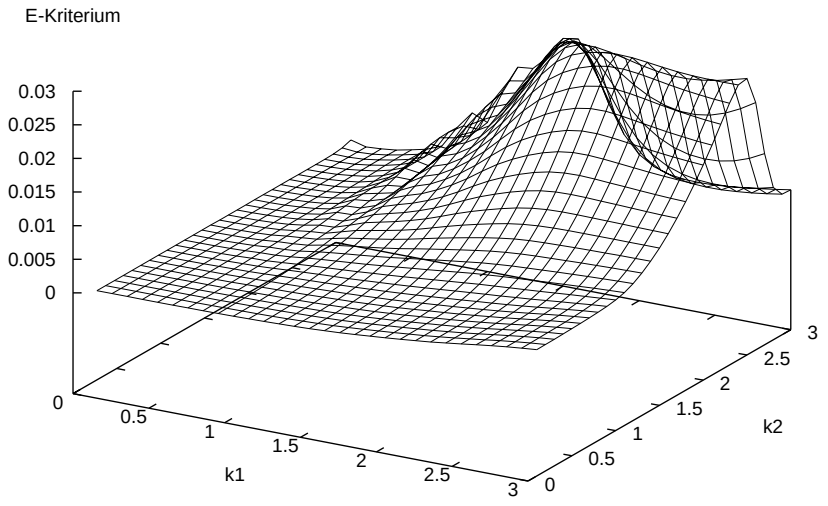
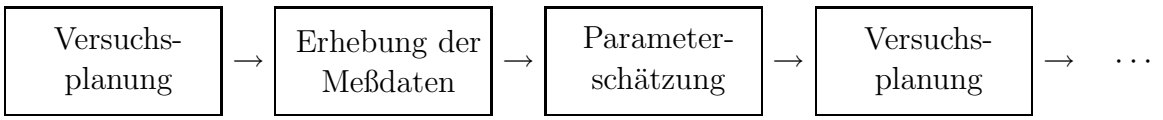


Abbildung 5.2: Werte des Gütekriteriums in Abhängigkeit der Parameter k_1 und k_2 für das Lotka-Volterra-Testbeispiel für einen Versuchsplan, der für mehrere Parameterwerte gleichzeitig optimiert wurde

zum Beispiel in Form von validierten Teilmodellen. Eine sequentielle Vorgehensweise wurde aber auch damals schon für die Versuchsplanung bei Modellfunktionen, die nichtlinear in den Parametern sind, vorgeschlagen.

Diese besteht aus der Wiederholung von Versuchsplanung, Erhebung der Meßdaten und Parameterschätzung



Nach der Planung und Durchführung der Versuche kann in manchen Fällen auch erst einmal eine Analysephase nötig sein, etwa dann, wenn das Modell nicht zu den erhobenen Meßdaten paßt. Dies kann Modelländerungen oder ähnliches beinhalten. Hierfür können zum Beispiel Methoden der Modelldiskriminierung eingesetzt werden, wie sie von Dieses [Die97] untersucht wurden.

Für die sequentielle Vorgehensweise verwenden wir folgenden Algorithmus (siehe auch Bauer et al. [BBKS99b] und Körkel et al. [KBBS99]):

Seien die für das Versuchsplanungsproblem notwendigen Größen, Funktionen und die Schranken an die Optimierungsvariablen definiert. Ferner seien Startwerte (p^0, s^0) für die Parameter und Parametrisierungsvariablen gegeben. Diese können zum Beispiel aus der Literatur aus früheren Messungen oder aus Schätzungen unter etwas veränderten Versuchsbedingungen herrühren. Eventuell können auch schon ${}^0N$ bereits zuvor durchgeführte Experimente etwa aufgrund einer vorhergehenden Modellvalidierung zur Verfügung stehen, aus denen bereits eine erste Schätzung der Parameter erfolgte. Diese Information wird bei der Erstellung der Versuchspläne berücksichtigt. Setze $l = 1$.

1. Plane ${}^lN$ neue Experimente, die die Information aus den ${}^0N, \ldots, {}^{l-1}N$ bereits durchgeführten Experimenten ergänzen.

2. Führe die ${}^lN$ neu vorgeschlagenen Experimente im Labor durch.

3. Schätze die Parameter $({}^lp, {}^ls)$ aus allen ${}^0N, \ldots, {}^lN$ bereits durchgeführten Experimenten. Als Startwert für die Parameterschätzung kann zum Beispiel der Wert aus der letzten Iteration $({}^{l-1}p, {}^{l-1}s)$ verwendet werden.

4. Falls das Zielfunktional der Versuchsplanung klein genug ist, beende den Algorithmus.

5. Ansonsten setze $l \to l + 1$ und gehe zu 1.

Bemerkung 5.5.1 (Ergänzende Experimente)
Die Versuchspläne für die unterschiedlichen Parameterwerte $({}^lp, {}^ls)$ sind nur jeweils lokal optimal. Werden im Lösungspunkt $({}^lp, {}^ls)$ neue Experimente geplant, so ist darauf zu achten, daß der neu entworfene Versuchsplan einen maximalen Zuwachs an Information

liefert. Hierfür werden die alten, bereits durchgeführten Experimente berücksichtigt. Dabei wird nur der Anteil der zu den neu zu planenden Experimenten der Jacobi-Matrix J in jedem Iterationsschritt des Optimierungsverfahrens für die Auswertung und Ableitungsgenerierung neu ausgewertet. Der Anteil der zu den bereits durchgeführten Experimenten gehörigen Jacobi-Matrix und ihrer Ableitungen ist konstant und wird nur einmal zu Beginn der Optimierung berechnet oder aus früheren Rechnungen übernommen.

Kapitel 6

Berechnung von Ableitungen der Lösungstrajektorie des DAE-Systems

Die numerische Lösung der in Kapitel 4 dargestellten Parameterschätzprobleme erfordert nicht nur die Generierung der Lösungstrajektorie des DAE-Systems, sondern auch deren Ableitungen nach den Parametern p und den Parametrisierungsvariablen s. Für die in Kapitel 5 dargestellten Versuchsplanungs-Optimierungsprobleme müssen zudem die Ableitungen der Lösungstrajektorie nach den Steuergrößen q sowie zweite gemischte Ableitungen nach Parametern, Parametrisierungsvariablen und Steuergrößen bereitgestellt werden. Sowohl bei der numerischen Lösung der Parameterschätzprobleme als auch (beziehungsweise erst recht) bei der Lösung der Versuchsplanungs-Optimierungsprobleme benötigt dies den größten Anteil an der Gesamtrechenzeit. Zudem müssen die Ableitungen mit der von der Optimierung geforderten Genauigkeit berechnet werden.

Auch bei der numerischen Lösung von Optimal-Steuerungsproblemen oder Feedback-Steuerungen bei DAE-Systemen benötigt man mindestens erste Ableitungen der Lösung nach den Steuergrößen. Eventuell ist eine Approximation zweiter Ordnung sinnvoll, wofür zweite Ableitungen nach den Steuergrößen benötigt werden.

Da wir im Optimierungskontext zur Sicherung der Konsistenz der Anfangswerte die relaxierte Formulierung für die DAE-Systeme verwenden, betrachten wir im folgenden Anfangswertprobleme für DAE-Systeme mit relaxierten algebraischen Gleichungen der Form (4.4). Der Übersichtlichkeit halber lassen wir den das jeweilige Experiment betreffenden Index l weg. Wir bezeichnen mit $x(t; \tau_j, s_j, p, q)$ die Lösung des Anfangswertproblems mit Anfangswert $x(\tau_j) = s_j$. Diese hängt in der Regel von den Anfangswerten s_j, den Parametern p und den Steuergrößen q ab, wofür wir im folgenden die effiziente Generierung der Ableitungen darstellen wollen.

6.1 Externe Numerische Differentiation

Sei $v = (s_j, p, q)$. Die Ableitung der Lösungstrajektorie $x(t; \tau_j, v)$ in Richtung $\Delta v \in \mathbb{R}^{n_v}$, $n_v = n_x + n_p + n_q$, kann mit Hilfe des Differenzenquotienten approximiert werden:

$$\frac{\partial x}{\partial v}(t; \tau_j, v) = \frac{x(t; \tau_j, v + \epsilon_v \Delta v) - x(t; \tau_j, v)}{\epsilon_v} + \mathcal{O}(\epsilon_v). \tag{6.1}$$

Trivialerweise würde man zunächst die Nominaltrajektorie und anschließend die variierten Trajektorien mit gestörten Anfangswerten, Parametern oder Steuergrößen auswerten. Probleme ergeben sich aber, wenn höhere Genauigkeiten für die Sensitivitäten erzielt werden sollen. Die Lösung $x(t; \tau_j, v)$ des DAE-Systems (4.2) wird mit Hilfe der BDF-Diskretisierung berechnet. Schon kleine Störungen in s_j, p oder q führen meist auf ein unterschiedliches Diskretisierungsgitter und Ordnungsschema. Als Faustregel kann man für die Genauigkeit der Sensitivitäten maximal die Hälfte der Genauigkeit von Nominal- und variierten Trajektorien erwarten. Um höhere Genauigkeiten für die Sensitivitäten zu erzielen, müssen Nominal- und variierte Trajektorien mit sehr hoher Genauigkeit berechnet werden, was zu einem enormen Anstieg der Rechenzeit führt.

Obiges Vorgehen wird als Externe Numerische Differentiation (END) bezeichnet, da die Differentiation außerhalb der Diskretisierung der Nominaltrajektorie vorgenommen wird.

Bereits Ortega und Rheinboldt [OR66] und später Gear und Vu [GV83] haben in ihren Artikeln darauf hingewiesen, daß bei der Ableitungsgenerierung darauf zu achten ist, daß der Lösungsoperator differenzierbar in den abzuleitenden Variablen ist. Für die Lösung und Ableitungsgenerierung bei Differentialgleichungssystemen favorisierten Gear und Vu damals ein BDF-Verfahren konstanter Schrittweite und Ordnung.

6.2 Interne Numerische Differentiation

Im folgenden stellen wir eine Methode dar, die durch Kopplung von Automatischer Differentiation der Modellfunktionen des DAE-Systems und Interner Numerischer Differentiation des BDF-Diskretisierungsschemas die Ableitungen mit der von der Optimierung geforderten Genauigkeit effizient berechnet.

Wir betrachten die Diskretisierung der Nominaltrajektorie als Folge von Abbildungen und leiten das Diskretisierungsschema selbst ab. Integratoren sind nur dann effizient, wenn sie die Ordnung und Schrittweite und die Lösung der Lineare-Algebra-Teilprobleme adaptiv während der Integration verändern. Allerdings sind die adaptiv erzeugten Komponenten auch abhängig von den Anfangswerten, den Parametern und den Steuergrößen. Ein veränderter Anfangswert zieht zum Beispiel in der Regel eine unterschiedliche Folge von Schrittweiten nach sich. Die durch die Diskretisierung erzeugte Folge von Abbildungen hängt dadurch nicht immer stetig von den Anfangswerten, den Parametern und Steuergrößen ab. Abhilfe schafft hier das Prinzip der Internen Numerischen Differentiation

(IND), dessen Idee und erste Realisierungen auf Bock [Boc81] zurückgehen. Wir leiten die adaptiv erzeugte Folge von Abbildungen ab, jedoch nicht die adaptiven Komponenten selbst. Dies bedeutet, daß alle adaptiv erzeugten Komponenten eingefroren werden, wie etwa

- die Fehlerkontrolle,
- die Ordnungs- und Schrittweitensteuerung
- und das Newton-Verfahren zur Lösung des impliziten Systems, inklusive
  - der Anzahl der Newton-Iterationen
  - und der Approximation der Iterationsmatrix J und ihrer Zerlegung.

Diese Vorgehensweise wenden wir nun auf das BDF-Verfahren an. Die Diskretisierung der Nominaltrajektorie im Schritt $n+1$ ist von der Form

$$F(x_{n+1},p,q) := \begin{pmatrix} A(t_{n+1},x_{n+1},p,q)\,\displaystyle\sum_{i=0}^{k}\alpha_i\,y_{n+1-i} + h\,f(t_{n+1},x_{n+1},p,q) \\ g(t_{n+1},x_{n+1},p,q) - \vartheta(t_{n+1})\,g(\tau_j,s_j,p,q) \end{pmatrix} = 0. \tag{6.2}$$

Die Ableitung zum Beispiel nach den Parametern p ergibt nach dem Satz über implizite Funktionen

$$\frac{\partial x_{n+1}}{\partial p} = -\left(\frac{\partial F}{\partial x}(x_{n+1},p,q)\right)^{-1}\frac{\mathrm{d}F}{\mathrm{d}p}(x_{n+1},p,q)$$

und somit bei eingefrorenem Diskretisierungsschema

$$\begin{aligned} \left(A_x(t_{n+1},x_{n+1},p,q)\,\frac{\partial x_{n+1}}{\partial p} + A_p(t_{n+1},x_{n+1},p,q)\right)\sum_{i=0}^{k}\alpha_i\,y_{n+1-i} & \\ +A(t_{n+1},x_{n+1},p,q)\sum_{i=0}^{k}\alpha_i\,\frac{\partial y_{n+1-i}}{\partial p} & \\ +h\,f_x(t_{n+1},x_{n+1},p,q)\,\frac{\partial x_{n+1}}{\partial p} + h\,f_p(t_{n+1},x_{n+1},p,q) =0 & \\ g_x(t_{n+1},x_{n+1},p,q)\,\frac{\partial x_{n+1}}{\partial p} + g_p(t_{n+1},x_{n+1},p,q) - \vartheta(t_{n+1})\,g_p(\tau_j,s_j,p,q) =0. & \end{aligned} \tag{6.3}$$

Betrachtet man andererseits die Variationsdifferentialgleichung für die Wronski-Matrizen $W_p = (W_p^{yT}, W_p^{zT})^T = ((\partial y/\partial p)^T, (\partial z/\partial p)^T)^T$

$$\begin{aligned} (A_x\,W_p + A_p)\,\dot{y} + A\cdot\dot{W}_p^y &= f_x\,W_p + f_p \\ 0 &= g_x\,W_p + g_p - \vartheta\,g_p(\tau_j) \end{aligned} \tag{6.4}$$

und wendet darauf die BDF-Diskretisierung an, so erhält man exakt die Gleichungen (6.3). Leitet man auch die Monitor-Strategie zur Lösung des impliziten Systems für die

Nominaltrajektorie ab, so ist die somit berechnete Lösung die exakte Ableitung der diskretisierten Lösungstrajektorie. Der Name „Interne Numerische Differentiation“ ist dadurch motiviert, daß die Differentiation innerhalb des Diskretisierungsschemas vorgenommen wird.

Allgemein schreiben wir die Variations-DAEs für die Ableitungen nach Anfangswerten, Parametern und Steuergrößen verkürzt als

$$A\,\dot{W}^y = \begin{pmatrix} -A_y\,\dot{y}+f_y & -A_z\,\dot{y}+f_z & -A_p\,\dot{y}+f_p & -A_q\,\dot{y}+f_q \end{pmatrix} \cdot \begin{pmatrix} & W & & \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{pmatrix}$$

$$0 = \begin{pmatrix} g_y & g_z & g_p & g_q \end{pmatrix} \cdot \begin{pmatrix} & W & & \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{pmatrix} - \vartheta(t) \begin{pmatrix} g_y(\tau_j) & g_z(\tau_j) & g_p(\tau_j) & g_q(\tau_j) \end{pmatrix},$$

mit

$$W(t;\tau_j) = \begin{pmatrix} W^y(t;\tau_i) \\ W^z(t;\tau_i) \end{pmatrix} = \begin{pmatrix} W^y_{s^y} & W^y_{s^z} & W^y_p & W^y_q \\ W^z_{s^y} & W^z_{s^z} & W^z_p & W^z_q \end{pmatrix} = \begin{pmatrix} \frac{\partial y}{\partial s_i^y} & \frac{\partial y}{\partial s_i^z} & \frac{\partial y}{\partial p} & \frac{\partial y}{\partial q} \\ \frac{\partial z}{\partial s_i^y} & \frac{\partial z}{\partial s_i^z} & \frac{\partial z}{\partial p} & \frac{\partial z}{\partial q} \end{pmatrix} \Bigg|_{(t;\tau_i,s_i,p,q)}$$

Für die Berechnung der Ableitungen in Richtung einer Matrix $WD = (WD_x^T, WD_p^T, WD_q^T)^T \in I\!R^{(n_x+n_p+n_q)\times n_{dir}}$ lösen wir direkt die Variations-DAE in $\widetilde{W} = WD \cdot W$

$$\begin{aligned} A\,\dot{\widetilde{W}}^y &= \begin{pmatrix} -A_y\,\dot{y}+f_y & -A_z\,\dot{y}+f_z & -A_p\,\dot{y}+f_p & -A_q\,\dot{y}+f_q \end{pmatrix} \cdot \begin{pmatrix} \widetilde{W} \\ WD_p \\ WD_q \end{pmatrix} \\ 0 &= \begin{pmatrix} g_y & g_z & g_p & g_q \end{pmatrix} \cdot \begin{pmatrix} \widetilde{W} \\ WD_p \\ WD_q \end{pmatrix} - \vartheta(t) \begin{pmatrix} g_y(\tau_j) & g_z(\tau_j) & g_p(\tau_j) & g_q(\tau_j) \end{pmatrix} \cdot WD, \end{aligned} \tag{6.5}$$

wofür nur noch die Berechnung von n_{dir} Richtungen nötig ist. Der Term $\begin{pmatrix} g_y(\tau_j) & g_z(\tau_j) & g_p(\tau_j) & g_q(\tau_j) \end{pmatrix} \cdot WD$, der aus der Ableitung der Relaxierung folgt, ist im gesamten Integrationsintervall konstant und muß nur einmal zu Beginn der Integration ausgewertet werden.

Der Diskretisierungsfehler der Lösung der Nominaltrajektorie ist nach Satz 2.1.3 von der Ordnung $\mathcal{O}(h^k)$ wie auch der Diskretisierungsfehler der Lösung der Variationsdifferentialgleichungen.

Bemerkung 6.2.1 (Direkte Berechnung von Richtungsableitungen)
Wendet man den reduzierten Ansatz aus Abschnitt 4.4 zur Lösung der Parameterschätzprobleme an, so ist es nicht nötig, alle Ableitungen nach Parametern und Parametrisierungsvariablen zu berechnen. Vielmehr werden nur die Richtungsableitungen

$$H_j \cdot {}^{j-1}H_\kappa,\ \kappa \in \{E,R\},\ j \geq 0 \ \text{ und } \ H_j \cdot {}^{j-1}H_P + H_j^p$$

$$\textit{mit } H_j = \frac{\partial h_j}{\partial s_j} = \frac{\partial y(\tau_{j+1};\tau_j,s_j)}{\partial s_j} \textit{ und } H_j^p = \frac{\partial h_j}{\partial p} = \frac{\partial y(\tau_{j+1};\tau_j,s_j)}{\partial p},$$

benötigt. Dies sind $n_{dir} = (n_x - n_{elim}) + n_p + 1$ *Richtungen.*

Wir lösen die Variations-DAE (6.5) in den $n_{dir} = (n_x - n_c) + n_p + 1$ *Richtungen*

$$WD = \begin{pmatrix} {}^{j-1}H_E & {}^{j-1}H_P & {}^{j-1}H_R \\ 0 & I_{n_p} & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Richtungsableitungen werden auch für einen reduzierten Ansatz bei Optimal-Steuerungsproblemen wie zum Beispiel in dem von Leineweber et al. [Lei99, BBLS99] entwickelten Code MUSCOD-II benötigt. In diesem Fall müssen Richtungsableitungen in Richtung von Anfangswerten und Steuergrößen berechnet werden.

Zur Berechnung der Sensitivitäten mit Hilfe der Techniken der IND ergeben sich zwei grundsätzliche Möglichkeiten.

6.2.1 Variierte Trajektorien

Die Wronski-Matrizen W können mit finiten Differenzen – analog zur Approximation bei der END (6.1) – approximiert werden. Hierfür lösen wir ein Anfangswertproblem für die Nominaltrajektorie und die variierten Trajektorien mit gestörten Anfangswerten, Parametern und Steuergrößen

$$\Delta v = \epsilon_v \cdot \begin{pmatrix} \Delta s_i \\ \Delta p \\ \Delta q \end{pmatrix}.$$

Differentiation der adaptiv erzeugten Lösung des diskretisierten Systems für die Nominaltrajektorie bedeutet, daß die variierten Trajektorien mit dem gleichen Diskretisierungsschema wie für die Nominaltrajektorie berechnet werden. Das Newton-Verfahren zur Berechnung des nichtlinearen diskretisierten Gleichungssystems (6.2) wird dabei ebenso abgeleitet, das heißt, daß die Monitor-Strategie auch auf die Lösung der diskretisierten Systeme für die variierten Trajektorien angewendet wird. Die Iterationsmatrix für das vereinfachte Newton-Verfahren wird sowohl für die Nominal- als auch die variierten Trajektorien verwendet.

Der Fehler bei der Berechnung der Sensitivitäten setzt sich zum einen aus dem Diskretisierungsfehler und zum anderen aus der Approximation mit finiten Differenzen zusammen

$$err = \mathcal{O}(TOL) + \mathcal{O}(\sqrt{\epsilon_{mach}}),$$

falls $\epsilon_v^2 = \epsilon_{mach}$, ϵ_{mach} die Maschinengenauigkeit.

Einen Nachteil bei der Approximation mit variierten Trajektorien stellt die Wahl von ϵ_v dar. Als Faustregel wird dafür die Wurzel aus der Maschinengenauigkeit angegeben

(falls $\|\Delta v\| \approx 1$ und $\|v\| \approx 1$), jedoch wählt man bei nichtlinearen Problemen etwa $\epsilon_v = 10 \cdot \sqrt{\epsilon_{mach}} \cdot \|v\|$ oder $100 \cdot \sqrt{\epsilon_{mach}} \cdot \|v\|$. Im Verlauf der Integration kann sich auch die Differenz zwischen Nominal- und den variierten Trajektorien ändern, woraus sich weitere Genauigkeitsverluste ergeben können. Ein Neustart der Integration mit angepaßtem ϵ_v sollte dann durchgeführt werden. Da für die variierten Trajektorie – bei verändertem ϵ_v – keine zurückliegenden Werte zur Verfügung stehen, muß die Integration gänzlich neu gestartet werden – mit den in Kapitel 3 dargestellten Nachteilen.

Ein Vorteil bei dieser Vorgehensweise ist, daß nur die Ableitungen der Modellfunktionen nach den Zustandsvariablen x bereitgestellt werden müssen. Da die Ableitungen nur für die Aufstellung der Iterationsmatrix des modifizierten Newton-Verfahrens benötigt werden und für das Newton-Verfahren eine gute Approximation der Jacobi-Matrix ausreicht, ergeben sich keine Genauigkeitsverluste für die Lösung der Nominaltrajektorie, wenn diese zum Beispiel mit finiten Differenzen approximiert werden. Auch werden keine zusätzlichen Ableitungen nach p oder q benötigt.

Die größten Einsparungen werden zum einen dadurch erzielt, daß sich der Integrations-Overhead für die variierten Trajektorien verringert, vor allem aber dadurch, daß die Iterationsmatrix für das Newton-Verfahren für die variierten Trajektorien nicht neu berechnet und zerlegt werden muß.

Aufwand pro BDF-Schritt:

Lösen des linearen Gleichungssystems: $(n_{dir} + 1) \cdot n_{Newton}$
Funktionsaufrufe der Modellgleichungen: $(n_{dir} + 1) \cdot n_{Newton}$,

wobei n_{Newton} die Anzahl der Iterationen des vereinfachten Newton-Verfahrens bezeichnet.

zusätzlich:
ca. alle 5 - 20 BDF-Schritte: Berechnung von A_x, f_x, g_x
ca. alle 5 - 10 BDF-Schritte: Zerlegung der Iterationsmatrix.

6.2.2 Variations-DAE

Die zweite Möglichkeit zur Generierung der Ableitungen ist, die Variationsdifferentialgleichungen direkt zu berechnen. Die Lösung der diskretisierten Gleichungen für die Variations-DAEs (6.5) erfordert das Lösen des linearen Gleichungssystems

$$\underbrace{\begin{pmatrix} \alpha_0 A + h\, f_y - h\, A_y\, \dot{y} & h f_z - h\, A_z\, \dot{y} \\ g_y & g_z \end{pmatrix}}_{= J} \cdot W_{v,n+1} = \begin{pmatrix} -h\left((f_p - A_p\, \dot{y})\, WD_p + (f_q - A_q\, \dot{y})\, WD_q - A\, c(W_v)\right) \\ -g_v + \vartheta(t_{n+1})\, g_v(\tau_j)\, WD \end{pmatrix}. \tag{6.6}$$

Der Term $c(W_v)$ beschreibt den konstanten Anteil der Ableitung des Korrektorpolynoms für die Wronski-Matrizen W_v analog zu $c = c(y)$ für die Nominaltrajektorie aus Gleichung

(2.12). Ableitungen der Modellfunktionen nach v beschreiben Ableitungen nach s_i, p oder q oder eine Kombination davon.

Gleichungssystem (6.6) ist das diskretisierte System für die Wronski-Matrix $W_v = \partial x/\partial v$, v eine Komponente von s_i, p oder q oder eine allgemeine Richtung. Die Matrix J entspricht dabei gerade der Jacobi-Matrix des nichtlinearen Gleichungssystems für die diskretisierte Nominaltrajektorie, ausgewertet am aktuellen Punkt x_{n+1}. Zur Lösung des Gleichungssystems (6.6) verwenden wir zwei unterschiedliche Ansätze:

Variations-DAE mit Newton-Verfahren

Zur Berechnung der unbekannten Größen $W_{v,n+1}$ leiten wir auch die Berechnung von x_{n+1} aus der Diskretisierung der Nominaltrajektorie mit Hilfe des Newton-Verfahrens ab. Wendet man auch auf das vereinfachte Newton-Verfahren

$$\begin{aligned} x_{n+1}^{(m+1)} &= x_{n+1}^{(m)} + \Delta x_{n+1}^{(m)}, \\ \Delta x_{n+1}^{(m)} &= -\tilde{J}^{-1}\, F(x_{n+1}^{(m)}, p, q),\ m = 1, \{2, \{3\}\}, \end{aligned}$$

F aus Formel (6.2), die Kettenregel an

$$\frac{\partial x_{n+1}^{(m+1)}}{\partial v} = \frac{\partial x_{n+1}^{(m)}}{\partial v} - \tilde{J}^{-1} \left(\frac{\partial F(x_{n+1}^{(m)})}{\partial x_{n+1}^{(m)}} \frac{\partial x_{n+1}^{(m)}}{\partial v} + \frac{\partial F(x_{n+1}^{(m)})}{\partial v} \right),\ m = 1, \{2, \{3\}\},$$

so erhält man (hinreichend oft Differenzierbarkeit vorausgesetzt) eine Newton-Iteration für die Wronski-Matrizen $W_{v,n+1} = \frac{\partial x_{n+1}^{(m)}}{\partial v}$ der folgenden Form

$$W_{v,n+1}^{(m+1)} = W_{v,n+1}^{(m)} - \tilde{J}^{-1} \left(J(x_{n+1}^{(m)})\, W_{v,n+1}^{(m)} + F_v(x_{n+1}^{(m)}) \right),\ m = 1, \{2, \{3\}\}. \tag{6.7}$$

Dabei ist die Ableitung von F nach $x_{n+1}^{(m)}$ gerade die Jacobi-Matrix ausgewertet am aktuellen Punkt $x_{n+1}^{(m)}$ und die Ableitung von F nach v ist die rechte Seite von (6.6)

$$\begin{aligned} \frac{\partial F(x_{n+1}^{(m)})}{\partial x_{n+1}^{(m)}} &= J(x_{n+1}^{(m)}), \\ \frac{\partial F(x_{n+1}^{(m)})}{\partial v} &= \begin{pmatrix} -h\left((f_p - A_p\,\dot{y})\, WD_p + (f_q - A_q\,\dot{y})\, WD_q\right) - A\,h\,c(W_v) \\ -g_v + \vartheta(t_{n+1})\, g_v(\tau_j)\, WD \end{pmatrix}. \end{aligned}$$

Gleichung (6.7) bedeutet, daß das Newton-Verfahren mit derselben genäherten Jacobi-Matrix $\tilde{J}$ und derselben Anzahl von Iterationsschritten auf das Gleichungssystem (6.6) angewandt wird. Die Ableitungen $J(x_{n+1}^{(m)}) \cdot W_{v,n+1}^{(m)}$ können dann effizient direkt über Richtungsableitungen berechnet werden, was insbesondere die Richtungsableitungen der Modellfunktionen f, g und A nach den Richtungen $W_{v,n+1}^{(m)}$ erfordert. Diese können zum Beispiel mit Hilfe finiter Differenzen approximiert werden, etwa

$$\frac{\partial f}{\partial x}(x_{n+1}^{(m)}) \cdot W_{v,n+1}^{(m)} \doteq \frac{f(v + \epsilon_v \Delta v) - f(v)}{\epsilon_v}$$

$$\text{mit } v = \begin{pmatrix} x_{n+1}^{(m)} \\ p \\ q \end{pmatrix} \text{ und } \Delta v = \begin{pmatrix} W_{v,n+1}^{(m)} \\ WD_p \\ WD_q \end{pmatrix}$$

und analogen Berechnungen für die Ableitungen der algebraischen Gleichungen g und die Richtungsableitungen von A. Dabei ist die Wahl von ϵ_v gerade bei unterschiedlichen Größenordnungen der einzelnen Komponenten von W_v und p (bzw. von q) sehr problematisch. Eventuell sind zusätzliche Auswertungen der Modellfunktionen für eine hinreichend genaue Approximation nötig. Eine Auswertung der Richtungsableitungen mit Hilfe Automatischer Differentiation (zum Beispiel ADIFOR, siehe hierzu Bischof et al. [BCC+92, BCK+98]) ist dem in den meisten Fällen vorzuziehen.

Zudem müssen pro BDF-Schritt die Ableitungen der Modellfunktionen nach den Komponenten von v bereitgestellt werden. Dies kann ebenso über finite Differenzen oder wiederum mit Hilfe der Automatischen Differentiation erfolgen. Dabei wird ausgenutzt, daß die Modellfunktionen nur von einem Teil der Komponenten von v explizit abhängen.

Die oben beschriebene Lösung des linearen Gleichungssystems mit Newton-Verfahren ist gerade dann sinnvoll, wenn nur wenige Richtungen (verglichen mit der Anzahl der Zustandsvariablen) berechnet werden sollen. Die Monitor-Strategie für die Nominaltrajektorie zur Reduzierung des Lineare-Algebra-Aufwands wird auf die Berechnung der Unbekannten der diskretisierten Variations-DAEs übertragen.

Aufwand pro BDF-Schritt:

Lösen des linearen Gleichungssystems:	$(n_{dir}+1)\cdot n_{Newton}$
Auswertung der Richtungsableitungen $J(x_{n+1})\cdot W^{(m)}_{v,n+1}$:	$n_{dir}\cdot n_{Newton}$
Auswertung der Modellgleichungen:	n_{Newton}
Auswertung der Ableitungen der Modellgleichungen nach v:	n_{Newton}

zusätzlich:
ca. alle 5 - 20 BDF-Schritte: Berechnung von f_x, g_x
ca. alle 5 - 10 BDF-Schritte: Zerlegung von J

Variations-DAE mit direktem Lösen des linearen Gleichungssystems

Da das Gleichungssystem für die Variations-DAEs nach Diskretisierung (6.6) linear in den Unbekannten $W_{v,n+1}$ ist, kann dieses alternativ auch direkt gelöst werden. Allerdings entspricht diese Vorgehensweise zunächst nicht mehr dem Prinzip der IND.

Wird das diskretisierte System F im Lösungspunkt des Newton-Verfahrens ausgewertet, so erhalten wir eine leicht gestörte Lösung

$$F(x^{(m)}_{n+1}, p, q) = \delta.$$

Vernachlässigen wir diesen Fehler δ, so führt die Ableitung der Berechnung von x_{n+1} gerade auf die direkte Lösung des Gleichungssystems (6.6). Dies erfordert zwar die Auswertung und Zerlegung der Jacobi-Matrix J in jedem BDF-Schritt. Andererseits erspart man sich die vielen Auswertungen der Modellfunktionen und der Ableitungen nach Parametern und Steuergrößen, da das Gleichungssystem direkt und nicht mit bis zu drei Newton-Iterationen gelöst wird.

Aufwand pro BDF-Schritt:

Berechnung der Jacobi-Matrix J:	1
Zerlegung der Jacobi-Matrix J:	1
Lösen des linearen Gleichungssystems:	$n_{dir} + n_{Newton}$
Auswertung der Ableitungen der Modellgleichungen:	n_{dir}
Auswertung der Modellgleichungen:	n_{Newton}

6.2.3 Aufwandsabschätzung der unterschiedlichen Varianten

Die Hauptrechenzeit liegt bei allen Varianten zum einen in der Linearen Algebra und zum zweiten in der Berechnung der Ableitungen der Modellfunktionen. Zur Beurteilung der unterschiedlichen Varianten untersuchen wir zunächst den Aufwand für die Lineare Algebra und zur Berechnung der Ableitungen:

Zerlegung von J:

LU-Zerlegung einer Matrix benötigt

$$\sum_{k=1}^{n-1} \sum_{i=k+1}^{n} \sum_{j=k+1}^{n} 1 \;=\; \frac{1}{3}n^3 - \frac{1}{2}n^2 + \frac{1}{6}n$$

Operationen (Multiplikationen und Divisionen).

Lösen eines linearen Gleichungssystems:

Die Lösung eines linearen Gleichungssystems mit bereits zerlegter Matrix $J = LU$ benötigt

$$\sum_{j=1}^{n} \sum_{i=1}^{j-1} 1 + \sum_{j=1}^{n} \left(\overbrace{1}^{\text{Div.}} + \overbrace{\sum_{i=j+1}^{n} 1}^{\text{Mult.}} \right) = \sum_{j=1}^{n} \sum_{i=1}^{n} 1 = n^2$$

Operationen.

Berechnung der Ableitungen der Modellfunktionen:

Die Ableitungen von f, g und A werden in der Regel entweder mit numerischen Differenzen berechnet oder mit Hilfe der Automatischen Differentiation.

Numerische Differenzen: $n_{Abl.} = n_x + 1$ Funktionsaufrufe (bzw. $n_p + 1$, $n_q + 1$)
Automatische Differentiation: $n_{Abl.} = n_{AD_x}$ (bzw. n_{AD_p}, n_{AD_q})

Berechnung der Richtungsableitungen $J \cdot W_v$:

Numerische Differenzen: $n_{Abl.} = n_{dir} + 1$ Funktionsaufrufe
Automatische Differentiation: $n_{Abl.} = n_{AD_{dir}}$

Bemerkung 6.2.2 (Automatische Differentiation)
Die Automatische Differentiation hat den Vorteil, daß der Benutzer keinerlei Ableitungsinformation bereitstellen muß und die berechneten Ableitungen trotzdem exakt ausgewertet werden, wohingegen die mit finiten Differenzen berechneten Ableitungen einen Fehler in der Größenordnung von in der Regel mindestens $\sqrt{\epsilon_{mach}}$ aufweisen. Vor allem bei Richtungsableitungen mit finiten Differenzen mit nur $n_{dir} + 1$ Funktionsaufrufen gehen oft mehrere Stellen an Genauigkeit verloren, da sowohl der Richtungsvektor als auch die abzuleitenden Größen oft von sehr unterschiedlicher Größenordnung sind.

Ein Nachteil der Automatischen Differentiation zum Beipiel mit ADIFOR ist, daß die Auswertung der Ableitungen teils deutlich aufwendiger ist als die Berechnung mit finiten Differenzen, also $n_{dir} + 1$ Funktionsaufrufe.

Ein weiteres Einsparpotential lassen Automatische Differentiations-Tools erwarten, die die speziellen Strukturen der abzuleitenden Modellfunktionen ausnutzen. Rücker [Rüc99] entwickelte und implementierte in seiner Diplomarbeit effiziente Techniken zur automatischen Differentiation für chemische Reaktionsgleichungen. Durch Ausnutzung der dabei auftretenden Strukturen sind die von ihm generierten ersten und zweiten Ableitungen deutlich schneller als die mit Hilfe von ADIFOR erzeugten.

Vergleicht man die unterschiedlichen Varianten zur Berechnung der Sensitivitätsmatrizen, so sieht man, daß bei der Lösung der diskretisierten Systeme mit Newton-Verfahren zwar nur selten die Ableitungen der Modellfunktionen ausgewertet und die Iterationsmatrix zerlegt werden muß. Jedoch muß pro zu berechnender Richtung n_{Newton} mal das lineare Gleichungssystem mit bereits zerlegter Iterationsmatrix gelöst und die rechte Seite im Newton-Verfahren ausgewertet werden. Im Fall des direkten Lösens der linearen Gleichungssysteme der diskretisierten Variations-DAE muß die Auswertung des Gleichungssystems mit bereits zerlegter Iterationsmatrix und die Auswertung der rechten Seite nur einmal pro zu berechnender Richtung durchgeführt werden; jedoch auf Kosten der Berechnung und Zerlegung der Iterationsmatrix, die pro BDF-Schritt einmal durchgeführt werden muß.

Sollen nur wenige Richtungsableitungen berechnet werden (verglichen mit der Anzahl der Zustandsvariablen), so ist es ratsam, die Berechnung der Sensitivitäten mit variierten Trajektorien oder mit der Variationsdifferentialgleichung mit vereinfachtem Newton-Verfahren für das Lösen des linearen Gleichungssystems durchzuführen. Die Monitor-Strategie wird in beiden Fällen sowohl für die Lösung des impliziten Systems für die Nominaltrajektorie als auch für die variierten Trajektorien beziehungsweise für die Variationsdifferentialgleichungen angewendet. Dies spart Auswertungen und Zerlegungen der Iterationsmatrix J. Im Fall der variierten Trajektorien müssen dafür die Ableitungen der Modellfunktionen nach v pro Newton-Iteration, also ein bis drei mal pro BDF-Schritt ausgewertet werden. Im Fall der Berechnung der Ableitungen mit Hilfe der Variationsdifferentialgleichungen müssen die Richtungsableitungen $J \cdot W_v$, somit insbesondere $f_x \cdot W_v, g_x \cdot W_v$ und $W_v^T \cdot A_x \cdot (\alpha_0\, y_{n+1} + h\, c(y))$ pro Newton-Iteration ausgewertet werden.

Der Unterschied zwischen variierten Trajektorien und Variationsdifferentialgleichung mit

vereinfachtem Newton-Verfahren liegt in der unterschiedlichen Approximation der Ableitungsmatrizen. Falls die Ableitungen mit finiten Differenzen berechnet werden, sind in beiden Fällen pro BDF-Schritt $(n_{dir} + 1) \cdot n_{Newton}$ Auswertungen der Modellfunktionen nötig. Im Fall der variierten Trajektorien ergibt sich ein zusätzlicher Fehler dadurch, daß die Wronski-Matrizen W_v über Differenzenquotienten approximiert werden. Dieser liegt in der Größenordnung von ϵ_v, eine Genauigkeit von $\sqrt{\epsilon_{mach}}$ kann – gerade bei Richtungsableitungen – oft nicht erreicht werden. Die Strategie, den Abstand zwischen Nominaltrajektorie und variierten Trajektorien zu kontrollieren und bei einer starken Abweichung gegebenenfalls den Wert ϵ_v zu ändern, führt zwar zu besseren Genauigkeiten für die Sensitivitäten, erfordert jedoch jeweils einen Neustart des BDF-Verfahrens. Im zweiten Fall (Berechnung der Variationsdifferentialgleichung mit vereinfachtem Newton-Verfahren mit numerischen Differenzen zur Berechnung der Richtungsableitungen $J \cdot W_v + f_v$) erhält man durch die Approximation der Richtungsableitungen mit finiten Differenzen einen zusätzlichen Fehler. Werden die Richtungsableitungen aber exakt berechnet, z. B. mit Hilfe der Automatischen Differentiation, so entfällt dieser zusätzliche Fehler.

Die Berechnung der Variations-DAE mit direktem Lösen des linearen Gleichungssystems (6.6) ist dann sinnvoll, wenn viele Richtungen (verglichen mit der Anzahl der Zustandsvariablen) berechnet werden sollen. Der Aufwand von einer Berechnung und Zerlegung der Jacobi-Matrix J pro BDF-Schritt ist dann meist geringer als die zusätzliche Auswertung der Ableitungen $J \cdot W_v$ beziehungsweise der Ableitungen der Modellfunktionen nach v in jeder Newton-Iteration, also n_{Newton} mal pro BDF-Schritt.

Nur im Fall der Generierung der Ableitungen mit Hilfe der Variations-DAEs, der Anwendung der Monitor-Strategie auch auf die Lösung der aus der Diskretisierung der Variations-DAEs resultierenden Gleichungssysteme (mit der gleichen Iterationsmatrix und der gleichen Anzahl von Newton-Iterationen) und falls die Ableitungen der Modellfunktionen exakt berechnet werden, ist die berechnete Lösung die exakte Ableitung der diskretisierten Lösung der Nominaltrajektorie. Dies wurde von Bock [Boc83] als der *analytische Grenzfall der IND* bezeichnet. In allen anderen Fällen treten zusätzliche Fehler auf, die aber in den meisten Anwendungen unterhalb der vom Benutzer geforderten Genauigkeit TOL liegen.

Bemerkung 6.2.3 (Kontinuierliche Lösungsdarstellung)
Wird die Lösung mit Hilfe des Interpolationspolynoms an Punkten zwischen zwei Diskretisierungspunkten ausgewertet, so ist der dabei auftretende Fehler (zumindest bei konstanter Schrittweite) asymptotisch kleiner als der Diskretisierungsfehler an den Gitterpunkten. Die BDF-Polynome stellen somit eine fehlerkontrollierte kontinuierliche Darstellung der Lösung des Anfangswertproblems bereit (siehe Bock und Schlöder [BS81]). Das Diskretisierungsschema des BDF-Verfahrens kann vom Gitter zur Auswertung von Innere-Punkt-Bedingungen oder der Modellantworten entkoppelt werden. Die Berechnung der Größen – und insbesondere auch der Ableitungen hiervon – erfordert nur eine Auswertung der Interpolationspolynome.

6.3 Generierung von zweiten Ableitungen

Für die im vorhergehenden Kapitel zur Versuchsplanung dargestellten Optimierungsprobleme müssen zusätzlich zweite gemischte Ableitungen der Lösungstrajektorie $x(t)$ nach Parametern, Anfangswerten und Steuergrößen bereitgestellt werden. Hierfür schreiben wir das diskretisierte und bereits nach den Parametern p differenzierte System (6.3) verkürzt als

$$G(x_{n+1}, \frac{\partial x_{n+1}}{\partial p}, p, q) = 0$$

und erhalten nach dem Satz über implizite Funktionen

$$\frac{\partial}{\partial q}\left(\frac{\partial x_{n+1}}{\partial p}\right) = -\left[\frac{\partial G}{\partial\left(\frac{\partial x_{n+1}}{\partial p}\right)}(x_{n+1}, \frac{\partial x_{n+1}}{\partial p}, p, q)\right]^{-1} \frac{\mathrm{d}G}{\mathrm{d}q}(x_{n+1}, \frac{\partial x_{n+1}}{\partial p}, p, q)$$

und somit

$$\begin{aligned}
&\left(\alpha_0 A + h\, f_y + A_y \sum_{i=0}^{k} \alpha_i y_{n+1-i} \,\middle|\, h f_z + A_z \sum_{i=0}^{k} \alpha_i y_{n+1-i}\right) \frac{\partial^2 x_{n+1-i}}{\partial q \partial p} \\
&+ \left[\left(\frac{\partial x}{\partial q}\right)^T A_{xx} \frac{\partial x}{\partial p} + A_{xq} \frac{\partial x}{\partial p} + A_{px} \frac{\partial x}{\partial q} + A_{pq}\right] \sum_{i=0}^{k} \alpha_i y_{n+1-i} \\
&+ \left(A_x \frac{\partial x}{\partial p} + A_p\right) \sum_{i=0}^{k} \alpha_i \frac{\partial y_{n+1-i}}{\partial q} + \left(A_x \frac{\partial x}{\partial q} + A_q\right) \sum_{i=0}^{k} \alpha_i \frac{\partial y_{n+1-i}}{\partial p} \\
&+ h \left(\frac{\partial x}{\partial q}\right)^T f_{xx} \frac{\partial x}{\partial p} + h\, f_{xq} \frac{\partial x}{\partial p} + h\, f_{px} \frac{\partial x}{\partial q} + h\, f_{pq} = 0 \\
&g_x \frac{\partial^2 x_{n+1-i}}{\partial q \partial p} + \left(\frac{\partial x}{\partial q}\right)^T g_{xx} \frac{\partial x}{\partial p} + g_{xq} \frac{\partial x}{\partial p} + g_{px} \frac{\partial x}{\partial q} + g_{pq} - \vartheta\, g_{pq}(\tau_j) = 0
\end{aligned} \tag{6.8}$$

Betrachten wir die Variations-DAE für die Ableitungen $W_{p,q} = ((W^y_{p,q})^T, (W^z_{p,q})^T)^T = ((\partial^2 y/\partial q \partial p)^T, (\partial^2 z/\partial q \partial p)^T)^T$

$$\begin{aligned}
&\left[W_q^T A_{xx} W_p + A_{xq} W_p + A_x W_{p,q} + A_{px} W_q + A_{pq}\right] \dot{y} \\
&+ (A_x W_p + A_p)\, \dot{W}^y_q + (A_x W_q + A_q)\, \dot{W}^y_p + A\, \dot{W}^y_{p,q} \\
&= W_q^T f_{xx} W_p + f_{xq} W_p + f_x W_{p,q} + f_{px} W_q + f_{pq} \\
0 &= W_q^T g_{xx} W_p + g_{xq} W_p + g_x W_{p,q} + g_{px} W_q + g_{pq} - \vartheta\, g_{pq}(\tau_j)
\end{aligned} \tag{6.9}$$

und approximieren die Ableitungen $\dot{y}, \dot{W}_p, \dot{W}_q$ und $\dot{W}_{p,q}$ mit Hilfe der BDF-Diskretisierung ($-\rho/h$ aus (2.17)), so erhalten wir – das gleiche Diskretisierungsschema vorausgesetzt – gerade die Ableitung des diskretisierten Gleichungssystems (6.2). Wie für den Fall der ersten Ableitungen auch, ist die Lösung der Variations-DAE (6.9) zweiter Ordnung gerade die Ableitung der mit Hilfe des Diskretisierungsverfahrens erzeugten Nominallösung nach Parametern und Steuergrößen.

Analog stellen wir Variations-DAEs zweiter Ordnung für die Ableitungen nach Anfangswerten und Steuergrößen auf

$$\begin{aligned} &\left[W_q^T A_{xx} W_s + A_{xq} W_s + A_x W_{s,q}\right] \dot{y} + \left(A_x W_q + A_q\right) \dot{W}_s^y + A \dot{W}_{s,q}^y \\ &= W_q^T f_{xx} W_s + f_{xq} W_s + f_x W_{s,q} \\ 0 &= W_q^T g_{xx} W_s + g_{xq} W_s + g_x W_{s,q} - \vartheta\, g_{xq}(\tau_j). \end{aligned} \tag{6.10}$$

Zusammen mit der Nominallösung und den Variations-DAEs erster Ordnung müssen $N = 1 + n_p + n_s + n_q + (n_p + n_s) \cdot (n_q + n_s)$ Systeme berechnet werden.

Für die Lösung der Sensitivitätsgleichungen zweiter Ordnung (6.9) und (6.10) können – wie für den Fall erster Ableitungen auch – die Sensitivitätsmatrizen mit finiten Differenzen approximiert werden oder die Variations-DAEs direkt gelöst werden. Werden die Sensitivitätsmatrizen zweiter Ordnung mit finiten Differenzen approximiert, so sind die so berechneten Ableitungen in der Regel nicht mehr so genau – außer bei Approximation höherer Ordnung, wofür aber sehr viele variierte Trajektorien berechnet werden müßten.

Diese Form der Approximation hat in einer Hinsicht einen entscheidenden Vorteil, da – außer zur Lösung der impliziten diskretisierten Systeme (6.2) für Nominal- und variierte Trajektorien – keinerlei Ableitungen der Modellfunktionen bereitgestellt werden müssen und für die zu berechnenden Ableitungen gute Näherungen ausreichen, da schlechte Approximationen nur zu einer schlechteren Konvergenz des Newton-Verfahrens führen, jedoch an sich keinen Einfluß auf die Genauigkeit von Nominaltrajektorie und Sensitivitätsmatrizen haben. Ist die Anzahl an Parametern und Steuergrößen allerdings relativ hoch (gemessen an der Anzahl der Zustandsvariablen), so müssen sehr viele variierte Trajektorien berechnet werden, das heißt man benötigt $N \cdot n_{Newton}$ Auswertungen der Modellfunktionen des DAE-Systems und genau so viele Berechnungen von linearen Gleichungssystemen mit bereits zerlegter Matrix.

Bei der direkten Lösung der Variations-DAEs zweiter Ordnung müssen die ersten und zweiten Ableitungen der Modellfunktionen bereitgestellt werden. Dies kann zum Beispiel mit finiten Differenzen geschehen, allerdings treten hierbei wiederum Probleme mit der Genauigkeit der approximierten Ableitungen auf. Ungenaue Approximationen der Ableitungen führen auf gestörte Systeme und somit auch zu ungenauen Approximationen der Sensitivitätsmatrizen. Gerade wenn auch Sensitivitäten zweiter Ordnung berechnet werden sollen, empfiehlt es sich, die benötigten Ableitungen der Modellfunktionen exakt zu berechnen, wie es zum Beispiel mit Hilfe der Automatischen Differentiation erreicht werden kann. Der Benutzer muß dabei keinerlei Ableitungsinformation bereitstellen.

Nach Diskretisierung der Variations-DAEs zweiter Ordnung erhalten wir ein lineares Gleichungssystem in den Variablen (bzw. Matrizen) $W_{p,q,n+1}$ bzw. $W_{s,q,n+1}$. Dies kann (wie im Falle der Variations-DAEs erster Ordnung) direkt oder unter Anwendung der Monitorstrategie gelöst werden. Bei nur wenigen Richtungen ($N \ll n_x$) scheint die Methode mit Newton-Verfahren effizienter, falls N größer als etwa $n_{Newton} \cdot n_x$ ist, so ist das direkte Lösen meist effizienter.

Bemerkung 6.3.1 (Automatische Differentiation bei Large-Scale-Systemen) *Mit der im Moment zur Verfügung stehenden Version von ADIFOR (Version 2.0, Revision D) von Bischof et al. [BCC+92, BCK+98] können nicht direkt die zweiten Ableitungen der Modellfunktionen generiert werden. Für die in der Versuchsplanung benötigten zweiten gemischten Ableitungen der Modellfunktionen werden zunächst mit Hilfe von ADIFOR die Ableitungen nach Zustandsvariablen und Parametern generiert und die von ADIFOR erzeugten Routinen dann ein zweites Mal nach Zustandsvariablen und Steuergrößen abgeleitet. Die mit Hilfe der Sparse-Version von ADIFOR erzeugten Routinen enthalten allerdings Funktionen, die nicht nochmals abgeleitet werden können, so daß auch für große Systeme die Dense-Version zur Generierung der Ableitungen der Modellfunktionen verwendet werden muß. Die Matrizen müssen anschließend ins Sparse-Format umgespeichert werden, um die Sparse-Lineare-Algebra-Löser verwenden zu können.*

Ein weiteres Problem, schon bei der Generierung der ersten Ableitungen, besteht darin, daß sich der Sparsity-Pattern der Matrizen beziehungsweise Tensoren bei den mit ADIFOR berechneten Ableitungen im Verlauf der Integration ändern kann. Eine Strategie, die Faktorisierung der Jacobi-Matrizen so lange wie möglich einzufrieren, kann somit nicht einfach angewendet werden. Diese würde gerade bei der direkten Lösung der linearen diskretisierten Gleichungssysteme für die Variations-DAEs nochmals Einsparpotential bedeuten, da sich die Iterationsmatrizen von einem BDF-Schritt zum nächsten in der Regel nur wenig ändern.

6.4 Weitere Untersuchungen und Implementierungen zur Sensitivitätenberechnung

Für die Generierung von Sensitivitätsmatrizen erster Ordnung bei steifen ODE- und DAE-Systemen stehen inzwischen einige Integratoren zur Verfügung. Diese basieren in der Regel auf einer Erweiterung von bereits existierenden Codes zur numerischen Lösung von ODE- und DAE-Systemen. Für die numerische Realisierung gibt es grundsätzlich drei Möglichkeiten: a) die Lösung mit Hilfe der adjungierten Gleichungen, b) die Methode der finiten Differenzen und c) die Lösung mit Hilfe der Variationsdifferentialgleichungen beziehungsweise Variations-DAEs.

Wir geben zunächst einen Überblick über die verschiedenen Programmpakete zur Berechnung von Sensitivitäten. Im anschließenden Kapitel vergleichen wir die wichtigsten Codes mit den in DAESOL entwickelten Methoden. Die Generierung von zweiten Ableitungen der Lösungstrajektorie wird unseres Wissens von keinem anderen Programmpaket unterstützt.

Sargent und Sullivan [SS78] verwenden die adjungierten Gleichungen zur Generierung der Gradienten für nichtlineare Optimierungsprobleme in ODE-Systemen. Die Nominaltrajektorie und die adjungierten Gleichungen werden dabei mit einem BDF-Verfahren diskretisiert. Von Bock [Boc87] wurde ein adjungiertes Schema für Runge-Kutta-Verfahren aufgezeigt und implementiert, das den Prinzipien der IND folgt.

Die Generierung der Sensitivitätsmatrizen mit Hilfe finiter Differenzen führt – bei Anwendung der Prinzipien der IND – auf die sukzessive Lösung von variierten Trajektorien bei eingefrorenem Diskretisierungsschema der Nominaltrajektorie (siehe Bock et al. [Boc81, Boc83, Boc87, BES88, Eic87]). Die von Schittkowski zur Verfügung gestellten Integratoren in den Parameterschätzpaketen EASYFIT und PDEFIT [Sch99a, Sch99b] basieren fast ausschließlich auf der Approximation der Sensitivitätsmatrizen mit finiten Differenzen, allerdings als Black-Box-Ansatz, was der *Externen Numerischen Differentiation* mit seinen Nachteilen bei der Genauigkeit der Approximationen und dem immensen Rechenaufwand entspricht. Lediglich ein explizites Runge-Kutta-Verfahren berechnet die Sensitivitätsmatrizen nach den Prinzipien der IND.

Støren und Hertzberg [SH99] entwickelten eine Erweiterung von DASSL [Pet82a, Pet91, BCP96], genannt DASSP, die genau dem oben beschriebenen Verfahren der Lösung der variierten Trajektorien bei eingefrorenem Diskretisierungsschema entspricht. Bei Tests mit der von Maly und Petzold [MP96] entwickelten Erweiterung von DASSL – DASSLSO – und derjenigen von Caracotsios und Stewart [CS85] – DDASAC – war DASSP gegenüber den anderen beiden Verfahren meist etwas genauer und schneller. Jedoch wurden die in DASSLSO und DDASAC benötigten Ableitungen der Modellfunktionen zur Aufstellung der Variations-DAEs mit finiten Differenzen ausgewertet.

Die meisten existierenden Integratoren verwenden die Lösung der Variations-DAEs zur Generierung der Ableitungen. Diese sind meist Erweiterungen von bereits bestehenden Codes zur Simulation. Dunker [Dun84] implementierte eine Erweiterung des von Sherman und Hindmarsh [SH80] entwickelten BDF-Verfahrens GEARS zur Lösung von ODE-Systemen mit Hilfe der Variationsdifferentialgleichung. Bilardello et al. [BJL$^+$93] präsentieren einen Aufsatz auf LSODI, einer Erweiterung von LSODE für differentialalgebraische Gleichungssysteme von Hindmarsh [Hin80].

Caracotsios und Stewart erweiterten das von Petzold entwickelte Programmpaket DASSL zur Sensitivitätsberechnung für DAE-Systeme (DDASAC) [CS85] und für partielle Differentialgleichungen gekoppelt mit algebraischen Gleichungen (PDASAC) [CS95]. Zur Lösung der linearen Gleichungssysteme in jedem Schritt wird ein Newton-Verfahren verwendet. Die Jacobi-Matrix, die bei partiellen Differentialgleichungen sehr groß sein kann, wird allerdings trotzdem in jedem Schritt berechnet und zerlegt.

Maly und Petzold [MP96] erweiterten DASSL für Sensitivitätsberechnungen bei kleinen Systemen (DASSLSO) und bei Large-Scale Systemen (DASPKSO). Letztere verwendet Krylov-Methoden für die Lineare-Algebra-Teilprobleme. Sie stellen zunächst auch ein Gesamtsystem aus Nominal- und Variationsdifferentialgleichungen auf mit Jacobi-Matrix

$$J_{ges} = \begin{pmatrix} J & & & \\ J_1 & J & & \\ \vdots & & \ddots & \\ J_{n_{dir}} & & & J \end{pmatrix},$$

wobei J die Jacobi-Matrix der diskretisierten Nominaltrajektorie beschreibt und $J_i, i = 1, \ldots, n_{dir}$, die Ableitungen der Variationsdifferentialgleichungen nach den Zustandsvaria-

blen x. Das Gesamtsystem wird mit einem Newton-Verfahren berechnet mit genäherter Jacobi-Matrix $\tilde{J}_{ges}$, bei der die Matrizen J_i vernachlässigt werden. In den Lineare-Algebra-Teilproblemen wird ausgenutzt, daß die Jacobi-Matrix $\tilde{J}_{ges}$ in der Diagonalen jeweils die gleichen Blöcke J hat. Diese Variante wurde aber in einer neueren Arbeit von Li und Petzold [LP99a] als weniger effizient beschrieben als die Berechnung von Ableitungen mit Hilfe der Variations-DAEs mit Newton-Verfahren, wie es zum Beispiel in der Erweiterung von DASSL von Feehery et al. [FTB97] implementiert ist. Deshalb wurde inzwischen auch letztere Variante von Li und Petzold [LP99a] in DASSLSO und DASPKSO implementiert. Allerdings wird dabei zwar der Fehler im Newton-Verfahren für die Wronski-Matrizen kontrolliert, jedoch nicht darauf geachtet, daß nach dem Prinzip der IND die Anzahl der Newton-Iterationen von Nominaltrajektorie und Variations-DAEs gleich sein muß.

Auch das Programmpaket MATLAB [GH93] verfügt über eine Sensitivitätsberechnung (siehe Edsberg und Wedin [EW95]). Es handelt sich dabei allerdings um ein BDF-Verfahren von konstanter Ordnung 3 und einer vereinfachten Schrittweitensteuerung, bei der nur jeweils Änderungen um den Faktor zwei vorgenommen werden.

Kapitel 7

Numerische Ergebnisse

Das Programmpaket DAESOL wurde entwickelt und implementiert zur Simulation und Ableitungsgenerierung bei DAE-Systemen vom Index 1 der Form (4.2).

Im speziellen wurde dabei auf die folgenden Aspekte Wert gelegt:

- **Konsistente Initialisierung:** Zur konsistenten Initialisierung der algebraischen Gleichungen wird zunächst ein Vollschritt-Newton-Verfahren verwendet. Falls dieses nicht konvergiert, werden die konsistenten Anfangswerte mit einem Homotopie-Verfahren mit adaptiver Schrittweitensteuerung berechnet. Der Benutzer kann dabei eine der beiden Standard-Homotopien (3.1a, 3.1b) verwenden oder selbst eine Homotopie bereitstellen.

  Im Optimierungskontext wird eine relaxierte Formulierung der algebraischen Gleichungen unterstützt.

- **Simulation:** Das Programmpaket verfügt über eine adaptive Ordnungs- und Schrittweitensteuerung, deren Fehlerschätzungen das tatsächliche nichtäquidistante Diskretisierungsgitter berücksichtigen. Eine spezielle Monitor-Strategie reduziert den Lineare-Algebra-Aufwand, insbesondere aber die Anzahl der Auswertungen der Ableitungen der Modellfunktionen des DAE-Systems, zur Lösung der nichtlinearen Gleichungssysteme.

  Zur Reduzierung des Aufwands in der Startphase des BDF-Verfahrens wurde ein Runge-Kutta-Starter implementiert, der „zurückliegende“ Werte der Ordnung 3 beziehungsweise 4 für einen anschließenden Neustart des BDF-Verfahrens bereitstellt.

- **Ableitungsgenerierung:** Für Optimierungsprobleme aus Parameterschätzung und Optimalsteuerung ist die Generierung von ersten Ableitungen der Lösungstrajektorie des DAE-Systems nach Anfangswerten und Parametern beziehungsweise nach Anfangswerten und Steuergrößen nötig. Bei Verwendung eines reduzierten Ansatzes auch die Generierung der Ableitungen nach allgemeinen Richtungen.

  Für die Optimierungsprobleme in der Versuchsplanung werden zusätzlich zweite gemischte Ableitungen nach Anfangswerten, Parametern und Steuergrößen bereitgestellt.

Zur Lösung der Lineare-Algebra-Teilsysteme stehen in DAESOL grundsätzlich zwei Versionen zur Verfügung: eine Dense-Version für kleinere Systeme mit Lineare-Algebra-Lösern aus den Programmpaketen LINPACK von Dongarra et al. [DBMS80] und LAPACK von Anderson et al. [ABB+95] und eine Sparse-Version für große, dünnbesetzte Systeme mit den Lineare-Algebra-Lösern MA48 von Duff und Reid [DR96] und UMFPACK von Davis und Duff [DD95]. Zudem hat der Benutzer die Möglichkeit, weitere Lineare-Algebra-Löser – etwa Block-Band-Löser – anzusteuern. Mit Hilfe eines Shell-Skripts wird für die spezielle Variante eine Bibliotheksroutine erzeugt. Insbesondere wurde auf eine modulare Implementierung der unterschiedlichen Teilprobleme – konsistente Initialisierung, Fehlerschätzung, Schrittweitensteuerung, Lösung der nichtlinearen Gleichungssysteme für die diskretisierten Nominalgleichungen, Generierung der Ableitungen – Wert gelegt. Eine ausführliche Dokumentation zu DAESOL findet der Leser in Bauer et al. [BBS99].

Nachfolgend zeigen wir numerische Ergebnisse für die oben angeführten Aspekte, insbesondere den Vergleich von DAESOL mit anderen Softwarepaketen, die diese Features auch unterstützen. Im Anschluß geben wir Ergebnisse für Optimal-Steuerungsprobleme zur Versuchsplanung in DAE-Systemen an. Anhand zweier Beispiele für chemische Reaktionen in einem Semi-Batch-Reaktor zeigen wir die erfolgreiche sequentielle Vorgehensweise aus Versuchsplanung und Parameterschätzung.

Alle Rechnungen wurden auf einer SGI O2, 180 MHz, mit einem R 5000 Prozessor auf einem IRIX 6.3 Betriebssystem durchgeführt. Die Programme wurden alle mit den SGI Fortran und C++−Compilern mit der Optimierungsstufe „−mips2 −O2“ compiliert. Wenn nicht anders angegeben, wurden die für Simulation und Ableitungsgenerierung benötigten Ableitungen der Modellfunktionen des DAE-Systems mit Hilfe Automatischer Differentiation bereitgestellt. Hierfür verwendeten wir das Tool ADIFOR (Version 2.0, Revision D) von Bischof et al. [BCC+92, BCK+98] in der Dense-Version.

7.1 Konsistente Initialisierung

Zur Lösung eines Anfangswertproblems für DAE-Systeme der Form (4.2) mit Anfangswert $y(t_0) = y_0$ müssen zunächst die Anfangswerte konsistent sein, das heißt insbesondere, daß $g(t_0, y_0, z_0, p, q) = 0$ gelten muß. Falls die algebraischen Gleichungen stark nichtlinear in z sind, so ist dies mit einem Vollschritt-Newton-Verfahren oft nicht mehr einfach möglich, beziehungsweise müssten die Startschätzungen für z_0 sehr nah bei den tatsächlichen konsistenten Anfangswerten $z(t_0)$ liegen.

Beispiel 7.1 (Batch-Destillationskolonne)
Batch–Destillation wird vor allem in der Spezialitäten– und pharmazeutischen Chemie verwendet, wo man an der Produktion von geringen Mengen (z. B. von Aromen) interessiert ist. Ein Stoffgemisch oder verunreinigte Stoffe sollen während des Batch–Prozesses in ihre verschiedenen Bestandteile zerlegt werden.

Man geht von einem Anfangsgemisch in Blase, Böden und Kondensator aus. Das Gemisch wird in der Blase erhitzt, der Dampf V steigt durch die Kolonnen (Böden) in den Konden-

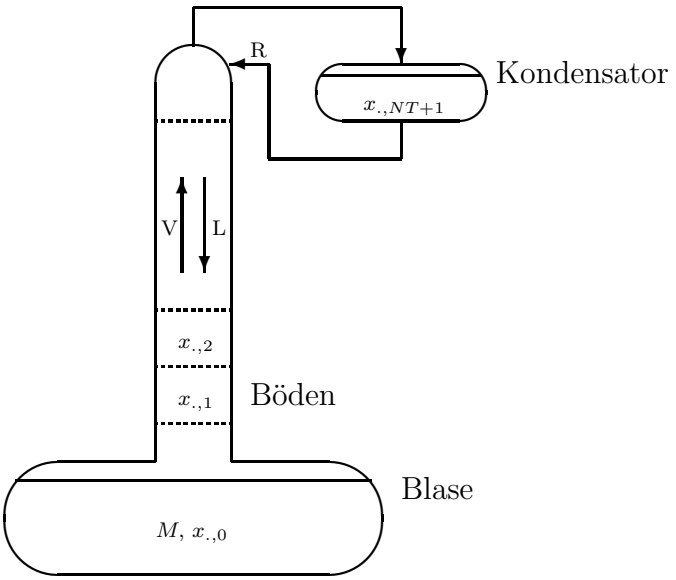


Abbildung 7.1: Modell einer Batch-Destillationskolonne

sator auf und wird dort kondensiert. Die kondensierte Flüssigkeit wird mit Rücklaufrate R in den obersten Boden zurückgegeben. Die gewünschten Stoffe können nacheinander aus dem Kondensator abgeführt werden (leichteste Komponente zuerst, dann zweitleichteste u.s.w.).

Das Modell beschreibt eine Batch–Destillation, bestehend aus einer Blase, $NT = 20$ Böden und vollständigem Kondensator. Es wird ein Gemisch aus $n_c = 10$ Komponenten betrachtet.

Das verwendete Modell wurde von Leineweber [Lei99] aufgestellt (siehe auch Farhat et al. [FCPD90]). Unter den Voraussetzungen eines idealen Gemischs, konstantem Druck, konstanter Verdampfungsraten und Flüssigkeitsströme erhält man die Differentialgleichungen für die Massenbilanzen

$$\frac{\mathrm{d}M}{\mathrm{d}t} = -\frac{V}{R+1}$$

$$\frac{\mathrm{d}(M\,x_{k,0})}{\mathrm{d}t} = -\frac{V}{R+1}\,x_{k,NT+1},\ k = 1,\ldots,n_c-1,$$

und die folgenden Erhaltungsgleichungen zwischen Blase, den Böden und Kondensator

$$K_k(T_l)\,x_{k,l} = \frac{R}{R+1}\,x_{k,l+1} + \frac{1}{R+1}\,x_{k,NT+1}, \quad k = 1,\ldots,n_c-1,\ l = 0,\ldots,NT,$$

$$\sum_{k=1}^{n_c} x_{k,l} = 1, \quad l = 0,\ldots,NT,$$

$$\sum_{k=1}^{n_c} K_k(T_l)\,x_{k,l} = 1, \quad l = 0,\ldots,NT.$$

Der Index $l = 0$ gibt die Variablen in der Blase an, der Index $l = NT + 1$ die zugehörigen Variablen im Kondensator.

Geht man von einem idealen Phasengleichgewicht in der Thermodynamik (Raoultsches Gesetz) aus, so sind die K-Werte von der Form

$$K_k(T) = \frac{P_k^s(T)}{P}, \quad \log_{10} P_k^s(T) = A_k - \frac{B_k}{C_k + T}. \tag{7.1}$$

A_k, B_k, C_k sind dabei die in Farhat et al. [FCPD90] gegebenen Antoine-Koeffizienten.

Dabei beschreiben

M	*die molare Masse des Gemischs in der Blase,*
V	*die Verdampfungsrate,*
L	*den abfließenden Flüssigkeitsstrom,*
R	*den Rücklauf,*
$x_{k,l}$	*den Molanteil der Flüssigkeit von Komponente k in Blase ($l = 0$), Böden ($l = 1, \ldots, NT$) und Kondensator ($l = NT + 1$),*
T_l	*die Temperatur in Blase ($l = 0$) und Böden ($l = 1, \ldots, NT$) und*
P	*den Druck.*

Die Anfangszusammensetzung des Gemischs in der Blase ist

$$x_{\cdot,0}(0) = (0.1, 0.3, 0.05, 0.04, 0.03, 0.08, 0.3, 0.03, 0.03, 0.1)^T. \tag{7.2a}$$

mit einer molaren Masse von

$$M(0) = 100\,[mol]. \tag{7.2b}$$

Das DAE-Modell ist vom Index 1. Wir nehmen eine konstante Rücklaufrate von $R = 10.0$ an, eine konstante Verdampfungsrate von $V = 110.0\,[mol/hr]$ und konstanten Druck $P = 1015.5\,[hPa]$. Wir erhalten $n_y = 10$ Differentialgleichungen für $M, x_{k,0}, k = 1, \ldots, n_c - 1$ und $n_z = 212$ algebraische Gleichungen für $x_{10,0}$, $T_l, l = 0, \ldots, NT$, $x_{k,l}, k = 1, \ldots, n_c - 1, l = 1, \ldots, NT$ und $x_{k,NT+1}, k = 1, \ldots, n_c$.

Eine Schwierigkeit bei der Simulation des Prozesses ist die Bestimmung konsistenter Anfangswerte für die algebraischen Variablen bei vorgegebenen Anfangswerten für die differentiellen Variablen (7.2a, 7.2b). Da die algebraischen Gleichungen sehr nichtlinear sind – die Temperatur in Blase und Böden tritt in Gleichung (7.1) im Exponenten auf – konvergiert ein Newton-Verfahren zur Bestimmung der konsistenten Startwerte nur, wenn die Schätzungen hierfür im lokalen Konvergenzbereich des Verfahrens, also sehr nah bei den wahren Werten liegen.

Zum Test von Verfahren zur konsistenten Initialisierung wurde zum einen der Integrator DASPK von Li und Petzold [LP99a, LP99b] verwendet. DASPK ist eine Weiterentwicklung des bekannten Programmpakets DASSL (siehe z. B. Brenan et al. [BCP96]). Der Algorithmus zur konsistenten Initialisierung – ein gedämpftes Newton-Verfahren – wird in Brown et al. [BHP98] und Li und Petzold [LP99a] beschrieben.

Weiterhin wurden das in DAESOL implementierte Homotopie-Verfahren zur konsistenten Initialisierung (siehe Abschnitt 3.1) und das Verfahren C05NCF aus der NAG-Programmbibliothek [NAG91] zum Lösen nichtlinearer Gleichungssysteme verwendet.

Für Tests zur konsistenten Initialisierung und dem möglichen Einzugsbereich der unterschiedlichen Verfahren wurden die folgenden Schätzungen für die Anfangswerte der algebraischen Variablen gewählt:

A: Wir nehmen die gleiche Zusammensetzung des Gemischs in den Böden und dem Kondensator an wie in der Blase vorgegeben. Die Temperatur wird auf 100 °C in der Blase und in allen Böden gesetzt.

B: Wie in A, setze jedoch die Temperatur auf 200 °C.

C: Wir geben zunächst Schätzungen für die Molanteile im Kondensator (leichteste Komponente gleich 1, alle anderen gleich 0) und für die Temperatur in Blase und dem obersten Kolonnenboden an ($T_0(0) = 250$ °C, $T_{NT}(0) = 200$ °C) und nehmen eine lineare Interpolation über die Kolonnenböden vor (Leineweber, private Mitteilung).

D: Setze die Werte aller algebraischen Variablen gleich 1.

Tabelle 7.1 zeigt die Anzahl der Aufrufe der algebraischen Gleichungen g zur konsistenten Initialisierung der Batch-Destillationskolonne mit Schätzungen für die Anfangswerte wie in Fall A-D gegeben. Die Jacobi-Matrix für das Newton-Verfahren wird mit Hilfe finiter Differenzen gebildet, wofür pro Auswertung n_z zusätzliche Aufrufe der algebraischen Gleichungen erforderlich sind. In dem vorliegenden Beispiel ist $n_z = 212$. Das Verfahren C05NCF gibt nur die Anzahl der Aufrufe des nichtlinearen Gleichungssystems aus.

Startschätzung	DAESOL	C05NCF	DASPK
A	$245 + 5 \cdot n_z = 1305$	—	—
B	$31 + 2 \cdot n_z = 455$	1368	—
C	$289 + 3 \cdot n_z = 925$	1145	—
D	$731 + 25 \cdot n_z = 6031$	—	—

Tabelle 7.1: Anzahl der Aufrufe der algebraischen Gleichungen zur konsistenten Initialisierung der Batch-Destillationskolonne mit Schätzungen für die Anfangswerte für die Fälle A-D.

Für die Generierung konsistenter Anfangswerte im Programmpaket DAESOL wurde die Standard-Homotopie H_1 (3.1a) verwendet. Man sieht, daß das implementierte Homotopie-Verfahren in DAESOL deutlich weniger Aufrufe benötigt und vor allem auch für Schätzungen, die weiter weg von den wahren Werten liegen, noch konvergiert. Das Verfahren in DASPK konvergiert für keinen der betrachteten Fälle A-D.

7.2 Lösung von Anfangswertproblemen

Nachfolgend wird das in Abschnitt 2.3 beschriebene Programmpaket DAESOL (♦) mit den folgenden Integratoren verglichen:

DASPK (□) von Li und Petzold [LP99a, LP99b] ist ein BDF-Verfahren variabler Ordnung und Schrittweite zur Lösung von Anfangswertproblemen bei impliziten DAEs der Form

$$F(t, y(t), \dot{y}(t)) = 0, \quad y(t_0) = y_0,\ \dot{y}(t_0) = \dot{y}_0.$$

Die Schrittweitensteuerung basiert allerdings zum Teil auf Approximationen auf äquidistantem Gitter. Zudem wird bei schlechter Konvergenz des Newton-Verfahrens die gesamte Funktionalmatrix neu ausgewertet und zerlegt, was im allgemeinen zu einer deutlich höheren Anzahl an Auswertungen der Ableitungen der Modellfunktion F führt als zum Beispiel in DAESOL oder VODE. Das Programmpaket löst DAEs vom Index 0 bis 2. Zur Berechnung konsistenter Anfangswerte muß ein Indexvektor initialisiert werden, der angibt, welche der Gleichungen beziehungsweise Variablen differentielle beziehungsweise algebraische sind.

Der Code ist eine Weiterentwicklung des bekannten Programmpakets DASSL von Petzold ([Pet82a, Pet91], siehe auch Brenan et al. [BCP96]).

VODE (×) von Brown, Byrne und Hindmarsh [BBH89, BHB98] ist ein BDF-Verfahren variabler Ordnung und Schrittweite zur Lösung von Anfangswertproblemen bei ODE-Systemen. Es verfügt über eine ähnliche Strategie zur Reduzierung des Lineare-Algebra-Aufwands wie die in Abschnitt 2.3.1 beschriebene Monitor-Strategie in DAESOL.

LIMEX (∗) ist ein Extrapolationsverfahren zur Lösung von Anfangswertproblemen bei DAEs in linear-impliziter Form

$$B(t, y)\,\dot{y}(t) = f(t, y), \quad y(t_0) = y_0 \tag{7.3}$$

vom Index 0 oder 1. Die Matrix $B(t, y) \in \mathbb{R}^{n_y \times n_y}$ kann regulär oder singulär sein.

Für die Tests wurde die Version 4.1A von Ehrig und Nowak [EN99] verwendet.

RADAU (○), RADAU5 (●) : Der Code RADAU [HW98] ist ein voll-implizites Runge-Kutta-Verfahren variabler Ordnung (5, 9 bzw. 13) mit 3, 5 bzw. 7 internen Stufen und mit Schrittweitensteuerung; RADAU5 [HW96a] ist ähnlich zu RADAU, aber von konstanter Ordnung 5. Eine Beschreibung beider Codes findet sich in dem Buch von Hairer und Wanner [HW96b]. Die Codes lösen Anfangswertprobleme für DAE-Systeme der Form (7.3) vom Index 0 bis 3.

Wenn nicht anders angegeben, wurde eine Referenzlösung für die nachfolgenden Beispiele mit DAESOL mit $TOL = ATOL = 1 \cdot 10^{-14}$ erzeugt. Wir vergleichen den Rechenaufwand (CPU-Zeit) der oben genannten Integratoren in Abhängigkeit von der erzielten

Genauigkeit. Für die Genauigkeit *acc* wird die angestrebte absolute Toleranz $ATOL$ mit berücksichtigt:

$$acc = \max_{i \in \{i\,:\,|y_{ref,i}| \geq ATOL_i\}} \frac{|y_{ref,i} - y_i|}{\max(|y_{ref,i}|, ATOL_i/TOL)}$$

Der Skalierungsmodus ist in allen betrachteten Beispielen wie in (2.28) angegeben. Für die Vergleiche beginnen wir mit einem Modell für einen Batch-Reaktor, das aus 6 differentiellen und 4 algebraischen Gleichungen besteht.

Beispiel 7.2 (Batch-Reaktor)
Biegler, Damiano und Blau [BDB86] stellten ein Modell für eine kinetische Reaktion in einem Batch-Reaktor auf. Lösungen hierzu finden sich auch bei Caracotsios und Stewart [CS85] und bei Maly und Petzold [MP96].

Die Hauptreaktion ist gegeben durch

$$HA + 2\,BM \rightarrow AB + MBMH,$$

AB ist dabei das gewünschte Produkt.

Die Reaktion wurde insgesamt mit den folgenden Reaktionsgleichungen beschrieben:

Langsame Reaktionen:

$$\begin{aligned} M^- + BM &\underset{k_2}{\overset{k_1}{\rightleftharpoons}} MBM^- \\ A^- + BM &\xrightarrow{k_3} ABM^- \\ M^- + AB &\underset{k_5}{\overset{k_4}{\rightleftharpoons}} ABM^- \end{aligned}$$

Schnelle Reaktionen (modelliert als Gleichgewichtsreaktionen):

$$\begin{aligned} MBM^- + H^+ &\underset{}{\overset{k_6}{\rightleftharpoons}} MBMH \\ A^- + H^+ &\underset{}{\overset{k_7}{\rightleftharpoons}} HA \\ ABM^- + H^+ &\underset{}{\overset{k_8}{\rightleftharpoons}} HABM\,. \end{aligned}$$

Die Reaktionen können mit Hilfe der folgenden 6 differentiellen und 4 algebraischen Gleichungen modelliert werden:

$$\begin{aligned} \dot{y}_1(t) &= -k_3\, y_2(t)\, y_8(t) \\ \dot{y}_2(t) &= -k_1\, y_2(t)\, y_6(t) + k_2\, y_{10}(t) - k_3\, y_2(t)\, y_8(t) \\ \dot{y}_3(t) &= k_3\, y_2(t)\, y_8(t) + k_4\, y_4(t)\, y_6(t) - k_5\, y_9(t) \\ \dot{y}_4(t) &= -k_4\, y_4(t)\, y_6(t) + k_5\, y_9(t) \\ \dot{y}_5(t) &= k_1\, y_2(t)\, y_6(t) - k_2\, y_{10}(t) \\ \dot{y}_6(t) &= -k_1\, y_2(t)\, y_6(t) + k_2\, y_{10}(t) - k_4\, y_4(t)\, y_6(t) + k_5\, y_9(t) \end{aligned}$$

$$
\begin{aligned}
y_7(t) &= -0.0131 + y_6(t) + y_8(t) + y_9(t) + y_{10}(t) \\
y_8(t)\,(k_7 + y_7(t)) &= k_7\, y_1(t) \\
y_9(t)\,(k_8 + y_7(t)) &= k_8\, y_3(t) \\
y_{10}(t)\,(k_6 + y_7(t)) &= k_6\, y_5(t)
\end{aligned}
$$

mit den Zustandsvariablen für die Beschreibung der Konzentrationen der folgenden Spezies (in [mol/kg])

$$
\begin{aligned}
y_1 &= [HA] + [A^-] & y_6 &= [M^-] \\
y_2 &= [BM] & y_7 &= [H^+] \\
y_3 &= [HABM] + [ABM^-] & y_8 &= [A^-] \\
y_4 &= [AB] & y_9 &= [ABM^-] \\
y_5 &= [MBMH] + [MBM^-] & y_{10} &= [MBM^-]
\end{aligned}
$$

und den Werten für die Reaktionsgeschwindigkeiten und Gleichgewichtskonstanten

$$
\begin{aligned}
k_1 &= 21.893 & k_5 &= 1.07 \cdot 10^9 \\
k_2 &= 2.14 \cdot 10^9 & k_6 &= 7.65 \cdot 10^{-18} \\
k_3 &= 32.318 & k_7 &= 4.03 \cdot 10^{-11} \\
k_4 &= 21.893 & k_8 &= 5.32 \cdot 10^{-18}.
\end{aligned}
$$

Die Anfangswerte sind gegeben durch

$$
\begin{aligned}
y_1(0) &= 1.5776 & y_6(0) &= 0.0131 \\
y_2(0) &= 8.32 & y_7(0) &= \frac{1}{2}\left(-k_7 + \sqrt{k_7^2 + 4\,k_7\,y_1(0)}\right) \\
y_3(0) &= 0 & y_8(0) &= y_7(0) \\
y_4(0) &= 0 & y_9(0) &= 0 \\
y_5(0) &= 0 & y_{10}(0) &= 0.
\end{aligned}
\tag{7.4}
$$

Das System ist vom Index 1.

Wir integrieren das System von $t_0 = 0$ *bis* $t_{end} = 10$. *Abbildung 7.2 zeigt den Verlauf der Lösung für die jeweiligen Spezies. Die Konzentrationen von* H^+, ABM^- *und* MBM^- *bleiben nahezu bei Null* ($\approx 10^{-10}$).

Für die folgenden Tests starten wir das System mit konsistenten Anfangswerten (7.4) und berechnen Lösungen für $TOL = 1/2^i \cdot 10^{-2}$, $i = 0, \ldots, 29$, $ATOL$ ist ein Vektor der Dimension $n_x = 10$ und berücksichtigt die unterschiedlichen Größenordnungen der jeweiligen Komponenten. Als Anfangsschrittweite setzen wir $h_0 = TOL \cdot 10^{-1}$ für die BDF-Verfahren und $h_0 = TOL$ für die Runge-Kutta-Verfahren RADAU und RADAU5 und für das Extrapolationsverfahren LIMEX.

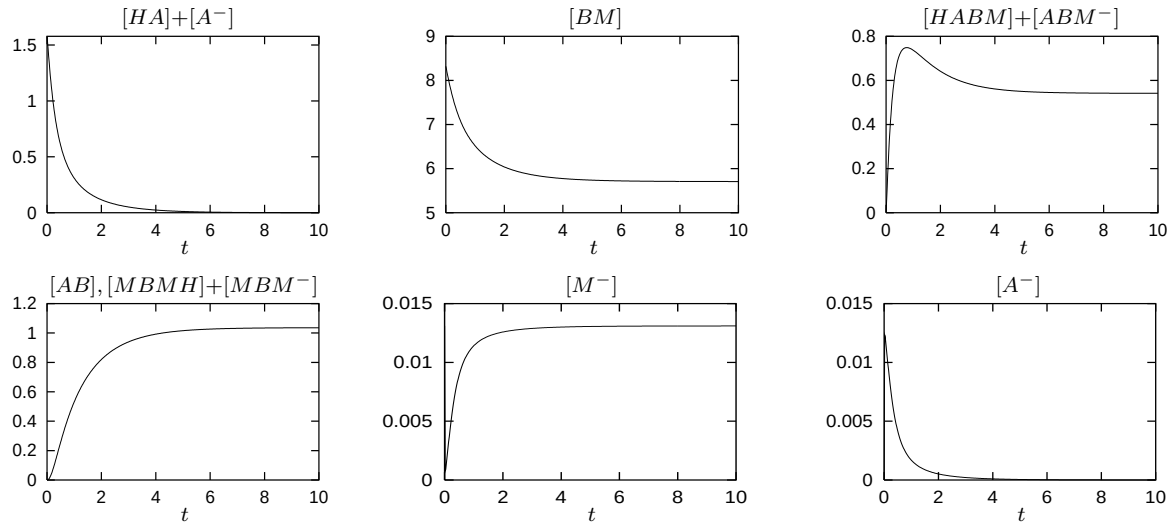


Abbildung 7.2: Lösungsverlauf der einzelnen Komponenten im Batch-Reaktor.

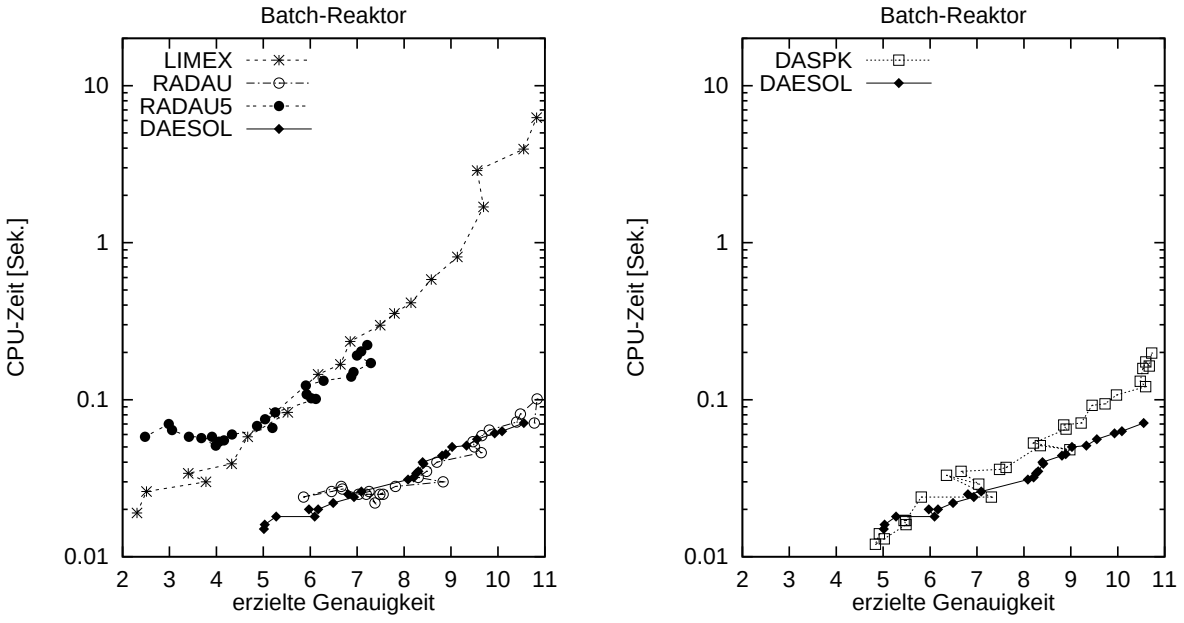


Abbildung 7.3: Benötigte CPU-Zeit für die jeweiligen Integratoren in Abhängigkeit von der erreichten Genauigkeit für das Modell des Batch-Reaktors.

		# f-Ausw.	# J-Ausw.	# J-Zerl.	CPU-Zeit [Sek.]
LIMEX	10^{-5}	1467	52	766	0.15
	10^{-8}	42600	271	19879	3.94
RADAU	10^{-5}	890	34	92	0.03
	10^{-8}	2343	55	236	0.07
RADAU5	10^{-5}	890	34	70	0.06
	10^{-8}	2297	55	130	0.14
DASPK	10^{-5}	708	59	59	0.04
	10^{-8}	1736	85	85	0.09
DAESOL	10^{-5}	606	29	67	0.03
	10^{-8}	1137	33	80	0.06

Tabelle 7.2: Vergleich der Integratoren für das Modell des Batch-Reaktors.

Abbildung 7.3 zeigt die benötigte CPU-Zeit in Abhängigkeit von der erreichten Genauigkeit. Der Hauptaufwand bei der Integration liegt in der Lösung der Lineare-Algebra-Teilprobleme und der Auswertung der Modellfunktionen des DAE-Systems. Tabelle 7.2 zeigt für $TOL = 1/2^{10} \cdot 10^{-2} \approx 1 \cdot 10^{-5}$ bzw. $TOL = 1/2^{20} \cdot 10^{-2} \approx 1 \cdot 10^{-8}$ die Anzahl an Auswertungen der Modellfunktionen des DAE-Systems (# f-Ausw.) und deren Ableitungen (# J-Ausw.) sowie die Anzahl der benötigten Zerlegungen (# J-Zerl.) für die unterschiedlichen Integratoren. Man sieht, daß die Monitor-Strategie in DAESOL gegenüber DASPK weniger als die Hälfte an Auswertungen der Ableitungen der Modellfunktionen benötigt. RADAU und RADAU5 verfügen zwar über ähnliche Strategien zur Reduzierung des Lineare-Algebra-Aufwands wie in DAESOL, benötigen jedoch für dieses Beispiel gerade bei höheren Genauigkeiten deutlich mehr Auswertungen der Ableitungen der Modellfunktionen und Zerlegungen der Jacobi-Matrix. Auch die Anzahl der Auswertungen der Modellfunktionen ist in DAESOL deutlich geringer als bei den übrigen Integratoren. Die Ableitungen der Modellfunktionen wurden analytisch bereitgestellt. Da VODE nur ODE-Systeme löst, konnte es bei den Vergleichen nicht berücksichtigt werden.

Beispiel 7.3 (Beispiel von Akzo Nobel)
Das im folgenden beschriebene Modell stammt aus der Beispielsammlung von Lioen und Swart des CWI in Amsterdam [LS]. Es wurde von der Firma Akzo Nobel aus den Niederlanden für Untersuchungen bei der chemischen Behandlung von Tumorgewebe aufgestellt.

Das Modell baut auf einer einfachen chemischen Reaktion auf

$$A + B \xrightarrow{k} C.$$

Ein radioaktiver Antikörper A reagiert mit einem Substrat B, dem Tumorgewebe. Der Antikörper ist dabei mobil, das Gewebe immobil. Wir erhalten ein 1D-Reaktions-Diffusions-System der Form

$$u_t = u_{xx} - k\,u\,v$$
$$v_t = -k\,u\,v.$$

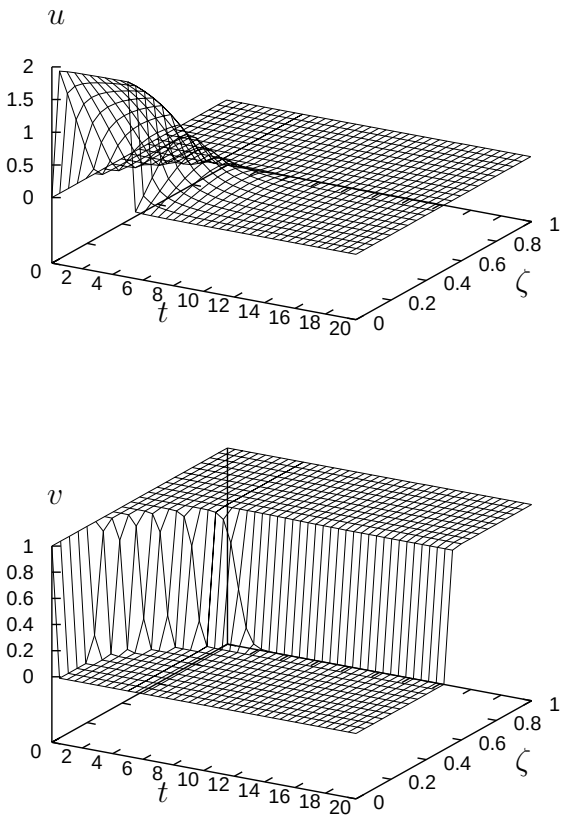


Abbildung 7.4: u und v als Funktionen der Zeit t und des Ortes ζ.

Ortsdiskretisierung mit Hilfe der Linienmethode ergibt das folgende ODE-System

$$\begin{aligned} f_{2j-1} &= \frac{(j\,\Delta\zeta - 1)^4}{c^2}\,\frac{y_{2j-3} - 2y_{2j-1} + y_{2j+1}}{\Delta\zeta^2} + \frac{(j\,\Delta\zeta - 1)^3}{c^2}\,\frac{y_{2j+1} - y_{2j-3}}{\Delta\zeta} - k\,y_{2j-1}\,y_{2j} \\ f_{2j} &= -k\,y_{2j-1}\,y_{2j} \end{aligned} \tag{7.5}$$

für $y \in \mathbb{R}^{2N}$, $j = 1, \ldots, N$, $\Delta\zeta = 1/N$, $t \in [0, T]$, $y_{-1}(t) = \phi(t)$, $y_{2N+1}(t) = y_{2N-1}(t)$, $y(0) = (0, 1, 0, 1, \ldots, 0, 1)^T$. *Bis zu einem gewissen Zeitpunkt wird der Antikörper A zugegeben. Die Infiltration* $\phi(t)$ *ist gegeben durch*

$$\phi(t) = \begin{cases} 2 & \text{für} \quad t \in [0, 5], \\ 0 & \text{für} \quad t \in (5, T],\ T \geq 5. \end{cases}$$

Für die Konstanten wurden die Werte $k = 100$, $c = 4$, $N = 200$ *und* $T = 20$ *angenommen. Wir integrieren das Sytem von* $t = 0$ *bis* $t_{end} = 20$. *Bei* $t = 5$ *ist aufgrund der Unstetigkeit der Infiltration ein weiterer Neustart nötig.*

Abbildung 7.4 zeigt u und v als Funktionen von t und ζ. Man sieht, daß eine Infiltration der chemischen Substanz A (lokal) B zerstört.

Die Jacobi-Matrix von System (7.5) hat Band-Struktur mit Bandbreite 5. Deshalb wird für die Tests in allen Integratoren ein Block-Band-Löser verwendet.

Abbildung 7.5 zeigt die benötigte CPU-Zeit der Integratoren in Abhängigkeit von der erreichten Genauigkeit. Die Jacobi-Matrizen wurden analytisch bereitgestellt.

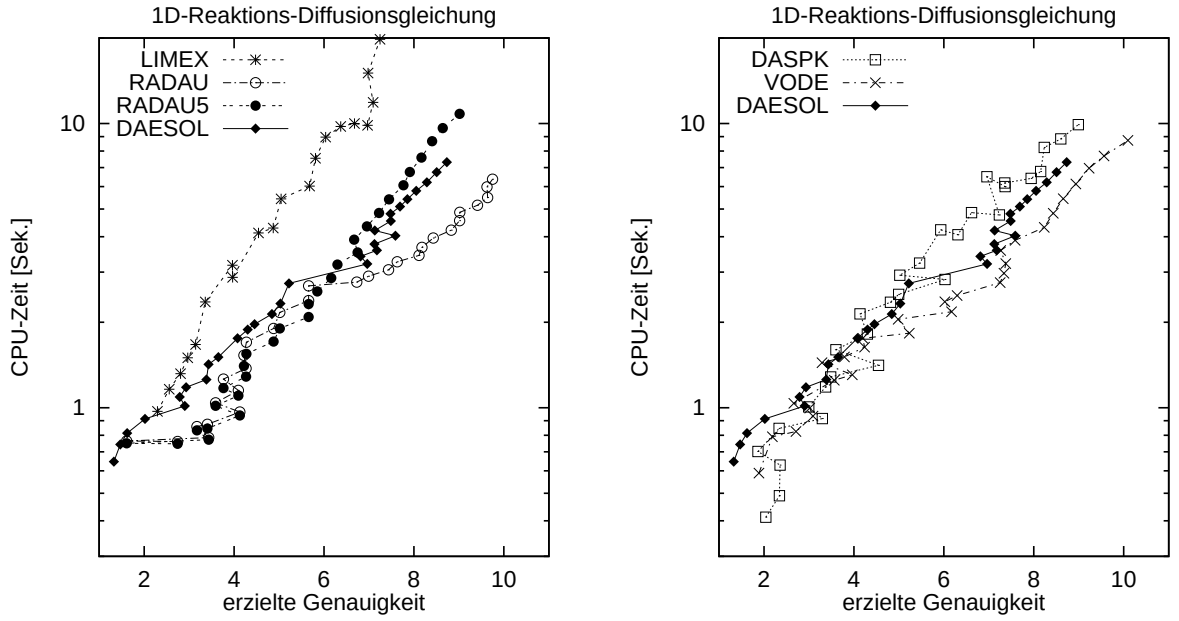


Abbildung 7.5: Benötigte CPU-Zeit für die jeweiligen Integratoren in Abhängigkeit von der erreichten Genauigkeit für das 1D-Reaktions-Diffusions-System.

Beispiel 7.4 (Batch-Destillationskolonne (Fortsetzung))
Als nächstes wollen wir die Integratoren für die bereits in Abschnitt 7.1 beschriebene Batch-Destillationskolonne testen. Es ist ein Modell bestehend aus 10 *differentiellen und* 212 *algebraischen Gleichungen. Die Jacobi-Matrix zur Lösung der Lineare-Algebra-Systeme ist zwar groß und dünn besetzt, weist allerdings keinerlei spezielle Strukturen auf. Die Lösung der Lineare-Algebra-Teilprobleme sollte deshalb mit einem Sparse-Löser wie etwa MA48 oder UMFPACK erfolgen. Da die anderen Integratoren allerdings nur dichte beziehungsweise Band-Matrizen unterstützen, werden die Vergleiche zwischen DAESOL und den übrigen Integratoren mit Lineare-Algebra-Lösern für dichte Systeme durchgeführt.*

Abbildung 7.6 zeigt die benötigte CPU-Zeit in Abhängigkeit von der erreichten Genauigkeit bei der Lösung des Batch-Destillationskolonnen-Modells. Wenn nicht anders angegeben, wurde für die Lineare Algebra ein dichter Löser verwendet. Die mit ($\diamond$) gekenn-

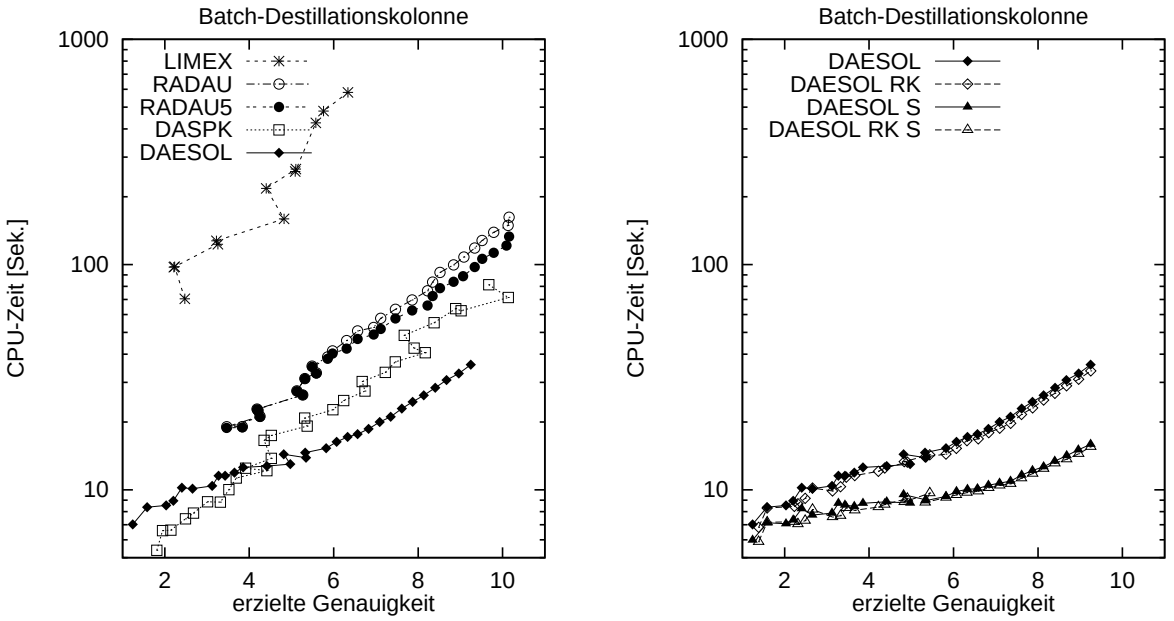


Abbildung 7.6: Benötigte CPU-Zeit für die jeweiligen Integratoren in Abhängigkeit von der erreichten Genauigkeit für die Batch-Destillationskolonne.

		# f-Ausw.	# J-Ausw.	# J-Zerl.	CPU-Zeit [Sek.]
LIMEX	10^{-5}	4145	299	2215	$4.26 \cdot 10^2$
	10^{-8}	—	—	—	—
RADAU	10^{-5}	1819	90	165	$3.54 \cdot 10^1$
	10^{-8}	4481	149	399	$8.36 \cdot 10^1$
RADAU5	10^{-5}	1819	90	163	$3.52 \cdot 10^1$
	10^{-8}	4481	149	295	$7.25 \cdot 10^1$
DASPK	10^{-5}	1098	66	66	$1.22 \cdot 10^1$
	10^{-8}	3294	114	114	$3.03 \cdot 10^1$
DAESOL	10^{-5}	1161	65	155	$(8.72)^*$ $1.26 \cdot 10^1$
	10^{-8}	2188	67	136	$(10.5)^*$ $1.87 \cdot 10^1$
DAESOL RK	10^{-5}	1144	66	152	$(8.39)^*$ $1.21 \cdot 10^1$
	10^{-8}	2207	68	134	$(10.2)^*$ $1.79 \cdot 10^1$

Tabelle 7.3: Vergleich der Integratoren für das Modell der Batch-Destillationskolonne. Die mit $(.)^*$ gekennzeichneten CPU-Zeiten geben die mit der Sparse-Version von DAESOL berechneten Ergebnisse an.

zeichnete Trajektorie gibt die CPU-Zeit für den Integrator DAESOL mit Runge-Kutta-Starter an, die für DAESOL mit S zusätzlich gekennzeichneten Trajektorien (▲) und (△) geben die CPU-Zeit der mit Hilfe der Sparse-Version von DAESOL (mit Sparse-Löser UMFPACK von Davis und Duff [DD95]) erzeugten Lösung ohne beziehungsweise mit Runge-Kutta-Starter an. Tabelle 7.3 zeigt für $TOL = 1/2^{10} \cdot 10^{-2} \approx 1 \cdot 10^{-5}$ bzw. $TOL = 1/2^{20} \cdot 10^{-2} \approx 1 \cdot 10^{-8}$ die Anzahl an Auswertungen der Modellfunktionen des DAE-Systems (# f-Ausw.) und deren Ableitungen (# J-Ausw.) sowie die Anzahl der benötigten Zerlegungen (# J-Zerl.) für die unterschiedlichen Integratoren. DASPK schneidet bei niedrigen Genauigkeiten sehr gut ab, allerdings nimmt der Aufwand mit zunehmender Genauigkeit stark zu. Die mit DAESOL mit der Sparse-Version erzeugten Lösungen sind in nahezu allen Fällen deutlich besser als die übrigen Integratoren. Mit Runge-Kutta-Starter ist DAESOL nochmals etwas schneller.

Beispiel 7.5 (Urethan-Reaktion)
Bei der Urethanreaktion handelt es sich um eine Simultan- und Konsekutivreaktion mit chemischem Gleichgewicht. Phenylisocyanat (A) und Buthanol (B) reagieren zu Urethan (C). Während der Hauptreaktion zum Urethan findet eine Folgereaktion zum Allophanat (D) statt, die je nach Verwendungszweck des entstehenden Produkts gewünscht ist oder auch nicht. Je nach verwendetem Katalysator kann das Phenylisocyanat zu Isocyanurat (E) trimerisieren.

Man erhält das folgende Reaktionsschema

$$\begin{aligned} \mathrm{A} + \mathrm{B} &\xrightarrow{k_1} \mathrm{C} \\ \mathrm{A} + \mathrm{C} &\underset{k_3}{\overset{k_2}{\rightleftharpoons}} \mathrm{D} \\ 3\,\mathrm{A} &\xrightarrow{k_4} \mathrm{E}. \end{aligned}$$

Ziel war es, die kinetischen Parameter zu bestimmen.

Die Reaktion wird in einem Rundkolben mit zwei Zuläufen im Semi-Batch-Betrieb durchgeführt. Im Rundkolben können die beiden Edukte Phenylisocyanat (A) und Butanol (B) vorgelegt werden, in den Zuläufen a und b Phenylisocyanat beziehungsweise Butanol, jeweils im Lösungsmittel Dimethylsulfoxid (DMSO) gelöst. Die Temperatur im Reaktor kann geregelt werden.

Die Reaktion wurde schon von mehreren Arbeitsgruppen untersucht (siehe z. B. Hugo [Hug97]), das verwendete DAE-Modell wurde gemeinsam von der BASF AG und dem IWR aufgestellt (siehe Bauer et al. [BHK+98, BKBS98]). Die Reaktionskinetik wurde unter Verwendung des Massenwirkungsgesetzes und der Annahme einer Arrhenius-Kinetik modelliert. Dies führt auf das folgende DAE-System

$$\begin{aligned} \dot{n}_3(t) &= V(t)\,(r_1(t) - r_2(t) + r_3(t)) \\ \dot{n}_4(t) &= V(t)\,(r_2(t) - r_3(t)) \\ \dot{n}_5(t) &= V(t)\,r_4(t) \end{aligned}$$

$$
\begin{aligned}
0 &= n_1(t) + n_3(t) + 2n_4(t) + 3n_5(t) - n_1(t_0) - n_{1ea}(t) \\
0 &= n_2(t) + n_3(t) + n_4(t) - n_2(t_0) - n_{2eb}(t) \\
0 &= n_6(t) - n_6(t_0) - n_{6ea}(t) - n_{6eb}(t)
\end{aligned}
$$

mit

$$n_3(t_0) = n_4(t_0) = n_5(t_0) = 0, \quad t_0 = 0.$$

Die Molzahlen $n_1(t), \ldots, n_5(t)$ der Stoffe A bis E und $n_6(t)$ des Lösungsmittels sind die (zeitlich variablen) Zustandsvariablen des DAE-Systems.

Die Ausdrücke $r_1(t), \ldots, r_4(t)$ beschreiben die Reaktionsgeschwindigkeiten der einzelnen Reaktionen, V ist das Reaktionsvolumen:

$$V(t) = \sum_{i=1}^{6} n_i(t)\frac{M_i}{\rho_i} \tag{7.6a}$$

$$r_1(t) = k_1(t)\,\frac{n_1(t)}{V(t)}\,\frac{n_2(t)}{V(t)} \tag{7.6b}$$

$$r_2(t) = k_2(t)\,\frac{n_1(t)}{V(t)}\,\frac{n_3(t)}{V(t)} \tag{7.6c}$$

$$r_3(t) = k_3(t)\,\frac{n_4(t)}{V(t)} \tag{7.6d}$$

$$r_4(t) = k_4(t)\,\left(\frac{n_1(t)}{V(t)}\right)^2 \tag{7.6e}$$

$$k_i(t) = f_{ref_i}\,\exp\left(-\frac{E_{ai}}{R}\left(\frac{1}{T(t)} - \frac{1}{T_{ref_i}}\right)\right), \quad i = 1, 2, 4 \tag{7.6f}$$

$$k_3(t) = \frac{k_2(t)}{k_c} \tag{7.6g}$$

$$k_c(t) = f_{c2}\,\exp\left(-\frac{dh_2}{R}\,\left(\frac{1}{T(t)} - \frac{1}{T_{c2}}\right)\right) \tag{7.6h}$$

$$n_{1ea}(t) = n_{1ea,0} \cdot inflow_a(t) \tag{7.6i}$$

$$n_{6ea}(t) = n_{6ea,0} \cdot inflow_a(t) \tag{7.6j}$$

$$n_{2eb}(t) = n_{2eb,0} \cdot inflow_b(t) \tag{7.6k}$$

$$n_{6eb}(t) = n_{6eb,0} \cdot inflow_b(t) \tag{7.6l}$$

Die Molmassen $M = (M_1, \ldots, M_6)^T$ und die Dichten $\rho = (\rho_1, \ldots, \rho_6)^T$ haben dabei die folgenden Werte

$$M = (0.11911, 0.07412, 0.19323, 0.31234, 3.11911, 0.07806)^T$$
$$\rho = (1095.0, 809.0, 1415.0, 1528.0, 1451.0, 1101.0)^T.$$

Unbekannte Parameter im Modell sind die Frequenzfaktoren f_{ref_i}, die Aktivierungsenergien E_{ai}, $i = 1, 2, 4$, und der Vorfaktor f_{c2} und die Reaktionsenthalpie dh_2 der Rückreaktion. Um eine starke Korrelation der Kinetikparameter – bei einer anschließenden Parameterschätzung – zu vermeiden, wird eine Umparametrisierung der Gleichungen (7.6f)

und (7.6g) mit Hilfe der Referenztemperaturen T_{ref_i} und T_{c2} vorgenommen. Diese geht auf Box zurück [Box60] und wird sehr oft zur Parameterschätzung in Kinetik-Modellen verwendet (siehe z. B. Agarwal und Brisk [AB85a, AB85b] und Doví et al. [DRAD94]). Die Frequenzfaktoren f_{ref_i} und der Vorfaktor der Rückreaktion f_{c2} sind dabei von den gewählten Referenztemperaturen abhängig. Wir wählen für die Referenztemperatur

$$T_{ref_i} = T_{c2} = 363.16[K].$$

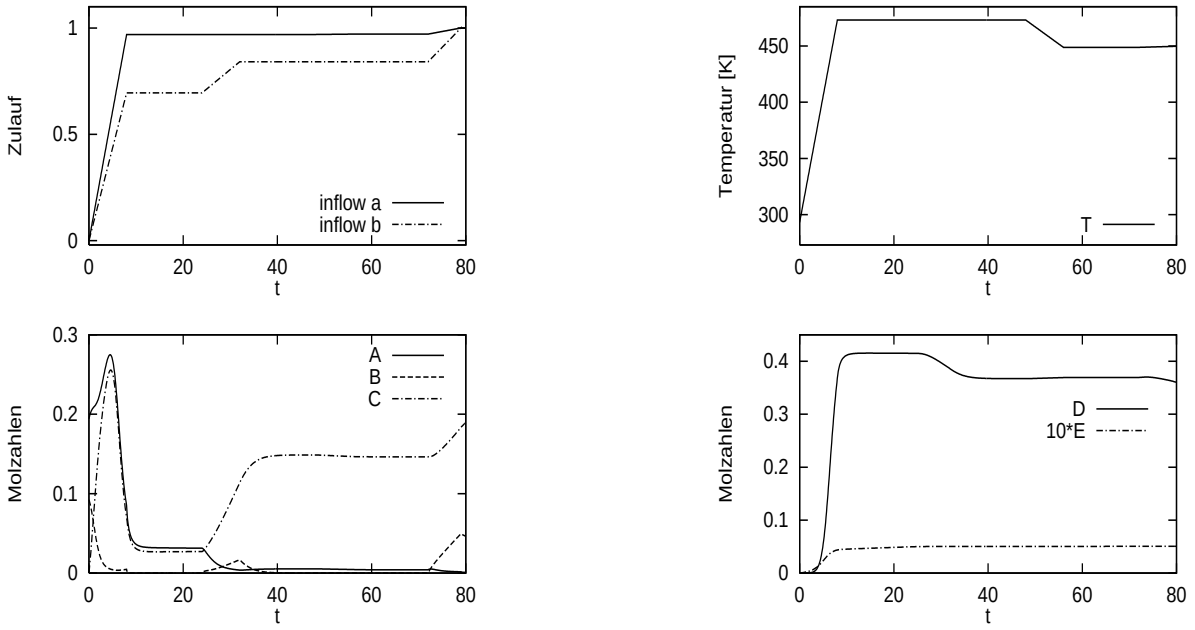


Abbildung 7.7: Dosier- und Temperaturprofile und zugehörige Lösungstrajektorien für die Urethan-Reaktion.

Verfahrenstechnische Komponenten des Modells sind die beiden Zuflüsse $inflow_a(t)$ und $inflow_b(t)$ und die Temperatursteuerung im Reaktor $T(t)$. zudem die Anfangszusammensetzung im Reaktor $n_1(t_0)$, $n_2(t_0)$ und $n_6(t_0)$ und in den beiden Zulaufgefäßen $n_{1ea,0}$ und $n_{6ea,0}$ beziehungsweise $n_{2eb,0}$ und $n_{6eb,0}$.

Die zeitlich variablen Steuerfunktionen $inflow_a(t)$, $inflow_b(t)$ und $T(t)$ werden parametrisiert und durch stückweise lineare und stetige Polynome ersetzt. Der Gesamtbeobachtungszeitraum beträgt 80 Stunden.

Für $n_1(t_0) = 0.194$, $n_2(t_0) = 0.0923$, $n_6(t_0) = 0.0960$, $n_{1ea,0} = 0.732$, $n_{6ea,0} = 0.124$, $n_{1eb,0} = 0.504$ und $n_{6eb,0} = 0.0$ und den wie in Abbildung 7.7 angegebenen Dosier- und Temperaturprofilen erhalten wir den folgenden Verlauf für die Lösungskomponenten $n_i(t)$, $i = 1, \ldots, 5$, $t \in [0, 80]$.

Abbildung 7.8 zeigt die Rechenzeiten der Integratoren in Abhängigkeit von der erzielten Genauigkeit. Aufgrund der Parametrisierung der Steuerfunktionen ist die rechte Seite

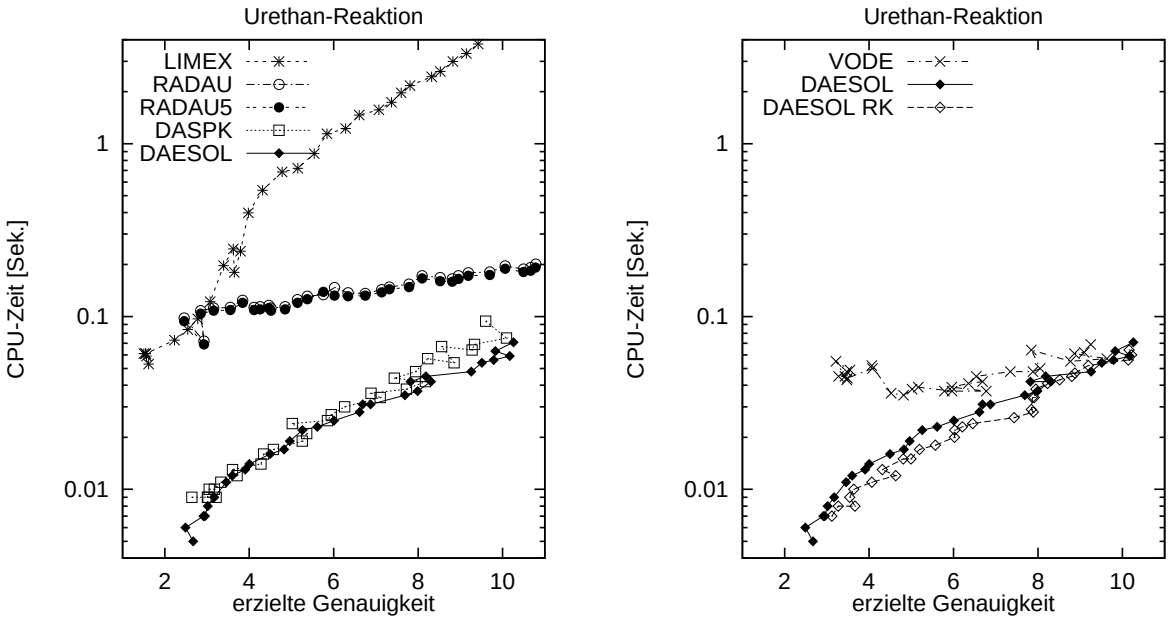


Abbildung 7.8: CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für das Urethan-Beispiel.

		# f-Ausw.	# J-Ausw.	# J-Zerl.	CPU-Zeit [Sek.]
LIMEX	10^{-5}	3278	395	2357	0.25
	10^{-8}	23922	1291	10143	1.46
RADAU	10^{-5}	6658	292	639	0.11
	10^{-8}	9445	381	961	0.17
RADAU5	10^{-5}	6658	292	639	0.11
	10^{-8}	9451	381	954	0.17
DASPK	10^{-5}	346	77	77	0.02
	10^{-8}	882	144	144	0.04
VODE	10^{-5}	979	102	235	0.04
	10^{-8}	1178	41	169	0.05
DAESOL	10^{-5}	469	15	61	0.02
	10^{-8}	1100	12	55	0.04
DAESOL RK	10^{-5}	448	15	42	0.02
	10^{-8}	1002	13	44	0.03

Tabelle 7.4: Vergleich der Integratoren für das Urethan-Beispiel.

des DAE-Systems an den Punkten $t = 8.0, 24.0, 32.0, 48.0, 56.0$ und 72.0 unstetig. An diesen Punkten ist jeweils ein Neustart erforderlich. Durch die vielen Neustarts ist DAESOL mit Runge-Kutta-Starter ($\lozenge$) deutlich schneller als ohne ($\blacklozenge$). DAESOL mit Runge-Kutta-Starter benötigt vor allem weniger Auswertungen der Modellfunktionen und der Ableitungen hiervon (siehe auch Tabelle 7.4). DASPK ist bei niedrigen Genauigkeiten ähnlich schnell wie DAESOL. Es benötigt im Durchschnitt weniger Auswertungen der Modellfunktionen, jedoch deutlich mehr Auswertungen von deren Ableitungen und deutlich mehr Zerlegungen der Jacobi-Matrix.

Beispiel 7.6 (2D-Konvektions-Diffusionsgleichung)
Wir betrachten die Gleichung

$$u_t = p_1\, u_{xx} + p_2\, u_{yy}$$

auf dem Gebiet $\Omega = [0,1] \times [0,1]$ *mit Dirichlet-Randbedingungen.*

Wir diskretisieren die partielle Differentialgleichung im Ort mit Hilfe der Linienmethode auf einem äquidistanten Gitter mit jeweils $M + 2$ *Gitterpunkten in* x- *und* y-*Richtung. Die Ortsableitungen werden mit zentralen Differenzen approximiert. Wir erhalten ein ODE-System im Innern des Gitters der Form*

$$\frac{\mathrm{d}u_{i,j}}{\mathrm{d}t} = p_1\, \frac{u_{i-1,j} + u_{i+1,j} - 2\,u_{i,j}}{\Delta x^2} + p_2\, \frac{u_{i,j-1} + u_{i,j+1} - 2\,u_{i,j}}{\Delta y^2}, \quad i,j = 1,\ldots,M,$$

mit $\Delta x = \Delta y = 1/(M+1)$. *Für die Ränder gelte* $u_{i,j} = 0$, $i = 0, M+1$, $j = 0, M+1$.

Wir lösen das System mit $M = 40$ *Diskretisierungspunkten. Als Startwerte wählen wir* $u_{i,j}(0) = 16\,(i\,\Delta x)\,(j\,\Delta y)\,(1 - i\,\Delta x)\,(1 - j\,\Delta y)$, $0 \le i,j \le M+1$, *und integrieren das System über einen Beobachtungszeitraum von* $[0, 100]$. *Für die Parameter wählen wir die Werte* $p_1 = p_2 = 1\,000$.

Das Modell wurde von Maly und Petzold [MP96] aufgestellt und als Benchmark-Beispiel insbesondere für die Ableitungsgenerierung verwendet (siehe auch Li und Petzold [LP99a]).

Das Beispiel kann als ODE- beziehungsweise DAE-System modelliert werden. Die Abbildungen 7.9 und 7.10 zeigen den Vergleich der unterschiedlichen Integratoren für das obige Beispiel. Auch bei diesem Beispiel sind die Rechenzeiten für DAESOL mit Runge-Kutta-Starter ($\lozenge$) signifikant niedriger als ohne ($\blacklozenge$), obwohl ein Neustart nur einmal zu Beginn der Integration nötig ist. Da die Ableitungen der Modellfunktionen konstant sind, ist bei den meisten Integratoren nur eine Auswertung hiervon nötig (siehe Tabelle 7.5). Die Tabelle zeigt, daß in DAESOL im Durchschnitt auch weniger Zerlegungen der Jacobi-Matrix nötig sind als bei den anderen Integratoren, mit Runge-Kutta-Starter werden noch weniger benötigt. Der Integrator VODE kann nur ODE-Systeme behandeln.

7.3 Ableitungsgenerierung

Für die in der Parameterschätzung und Optimierung benötigten Ableitungen der Lösungstrajektorie des DAE-Systems wurden in DAESOL effiziente Methoden nach den Techniken der IND entwickelt und implementiert. Je nach Größe des DAE-Systems und der

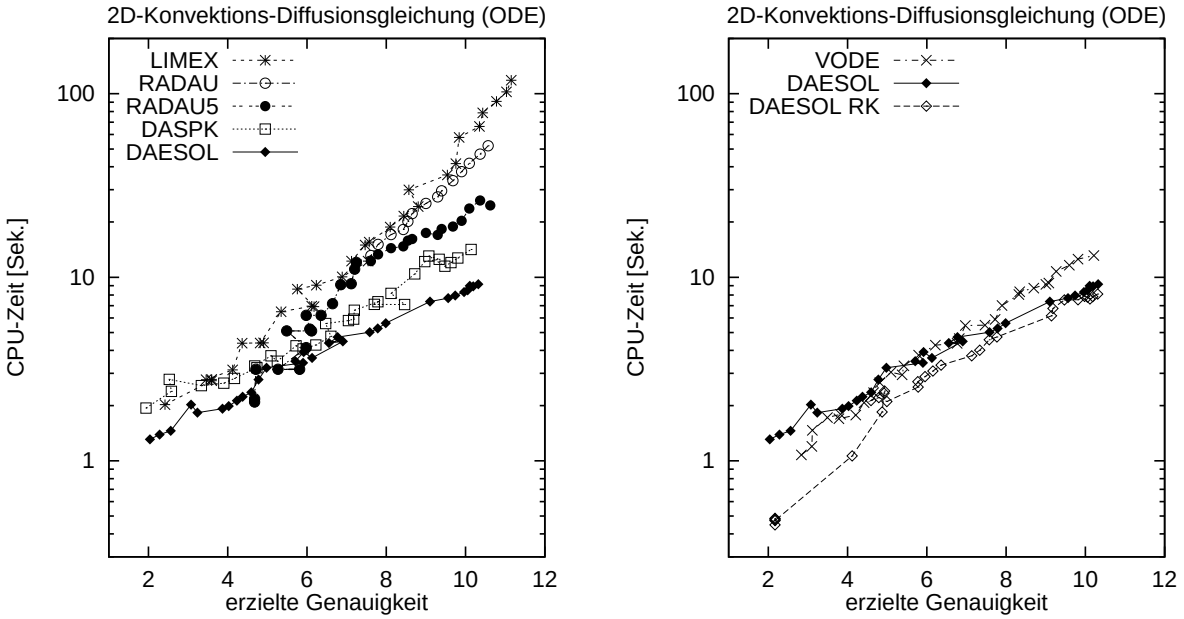


Abbildung 7.9: CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die Konvektions-Diffusionsgleichung modelliert als ODE-System.

		# f-Ausw.	# J-Ausw.	# J-Zerl.	CPU-Zeit [Sek.]
LIMEX	10^{-5}	37	1	19	6.9
	10^{-8}	176	1	63	24.3
RADAU	10^{-5}	31	1	6	6.2
	10^{-8}	96	1	20	20.2
RADAU5	10^{-5}	31	1	6	6.2
	10^{-8}	96	1	15	15.8
DASPK	10^{-5}	56	9	9	4.2
	10^{-8}	128	11	11	7.1
VODE	10^{-5}	43	1	9	3.0
	10^{-8}	124	2	17	7.0
DAESOL	10^{-5}	80	1	7	2.8
	10^{-8}	214	1	8	5.3
DAESOL RK	10^{-5}	74	1	5	2.2
	10^{-8}	195	1	7	4.7

Tabelle 7.5: Vergleich der Integratoren für die Konvektions-Diffusionsgleichung modelliert als ODE-System.

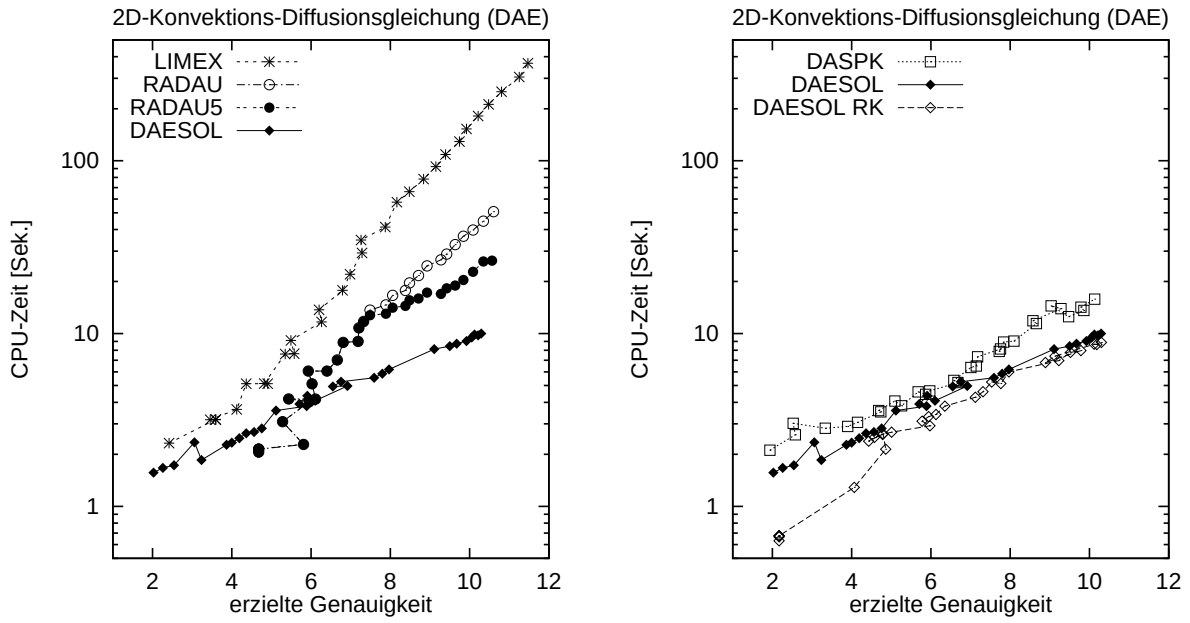


Abbildung 7.10: CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die Konvektions-Diffusionsgleichung modelliert als DAE-System.

Anzahl der zu berechnenden Richtungen ist zum einen die direkte Lösung der linearen Gleichungssysteme für die diskretisierten Sensitivitätsgleichungen sinnvoller oder aber die Anwendung des vereinfachten Newton-Verfahrens zur Lösung des impliziten nichtlinearen Gleichungssystems aus der Nominaltrajektorie auch für die Sensitivitätsgleichungen. Im folgenden werden wir die obigen Varianten kurz die *direkte* beziehungsweise *iterative Methode* nennen.

DDASAC (+) von Caracotsios und Stewart [CS85], eine Erweiterung von DASSL von Petzold [Pet82a] (siehe auch Brenan et al. [BCP96]) stellt nur die direkte Methode zur Verfügung, so daß die Jacobi-Matrix des vereinfachten Newton-Verfahrens aus der Nominaltrajektorie in jedem BDF-Schritt neu berechnet und zerlegt werden muß, was bei großen Systemen sehr aufwendig sein kann.

DASPK (□) von Li und Petzold [LP99a] stellt ähnliche Varianten wie in DAESOL zur Ableitungsgenerierung zur Verfügung: die direkte und die iterative Methode. Bei der iterativen Methode wird zwar der durch den vorzeitigen Abbruch im Newton-Verfahren gemachte Fehler kontrolliert, es wird allerdings nicht darauf geachtet, daß – im Sinne der IND – das vereinfachte Newton-Verfahren aus der Nominaltrajektorie abgeleitet werden muß. Zusätzlich stellt DASPK noch die von Maly und Petzold [MP96] eingeführte Variante zur Ableitungsgenerierung zur Verfügung: es wird das Gesamtsystem aus Nominaltrajektorie und Variations-DAEs gelöst. Dabei wird ausgenutzt, daß die Jacobi-Matrix des Gesamtsystems in der Diagonalen wiederholt die gleichen Blöcke hat. Das Verfahren (im folgenden kurz *simultane Methode* genannt)

entspricht den Prinzipien der IND, ist aber in den meisten Fällen aufwendiger als die beiden obigen Varianten.

Für die nachfolgenden Beispiele wurden für DAESOL und DASPK die Ableitungen mit allen Varianten berechnet, für die Tests wurde die jeweils schnellste verwendet. Falls das DAE-System relativ klein ist und die Auswertung und Zerlegung der Jacobi-Matrix J aus (2.14) (bzw. eine ähnliche Jacobi-Matrix für DASPK) nicht zu aufwendig und gleichzeitig relativ viele Richtungsableitungen (im Vergleich zur Anzahl der Zustandsvariablen) zu berechnen sind, so ist meist die direkte Methode schneller. DDASAC wird dann bei den Vergleichen mitberücksichtigt. Ist hingegen das DAE-System groß und sollen nur wenige Ableitungen berechnet werden, so ist in der Regel die iterative Methode vorzuziehen. DASPK arbeitet in letzterem Fall manchmal auch besser mit der simultanen Methode. Allerdings wurde von Li und Petzold [LP99a] keine Faustregel angegeben, in welchen Fällen welche Methode vorzuziehen ist, so daß vor Anwendung des Programms im Optimierungskontext zunächst immer beide Methoden getestet werden müssen.

Beispiel 7.7 (Batch-Reaktor (Fortsetzung))
Für das im vorherigen Abschnitt 7.2 dargestellte Beispiel des Batch-Reaktors sollen die Ableitungen nach den 8 *Parametern* $k_1, \ldots, k_8$ *und nach allen Anfangswerten der Zustandsvariablen berechnet werden. Sowohl für DAESOL als auch für DASPK ist das direkte Lösen der linearen Gleichungssysteme am schnellsten.*

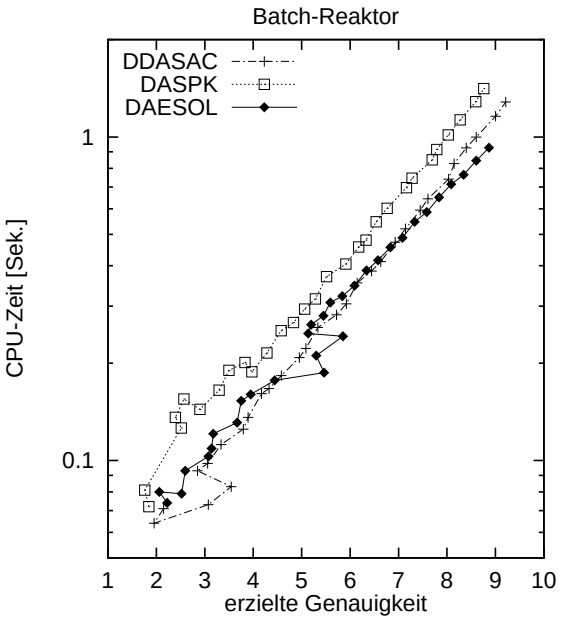


Abbildung 7.11: Benötigte CPU-Zeit für die jeweiligen Integratoren in Abhängigkeit von der erreichten Genauigkeit für die Ableitungsgenerierung für das Modell des Batch-Reaktors.

Abbildung 7.11 zeigt die CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für

die Ableitungen. DDASAC und DAESOL sind in etwa gleich schnell, DASPK hingegen benötigt etwas mehr Rechenaufwand.

Beispiel 7.8 (Batch-Destillationskolonne (Fortsetzung))
Für das in Abschnitt 7.1 dargestellte Modell der Batch-Destillationskolonne sollen die Ableitungen nach den Steuergrößen R, V und P und nach den Anfangswerten der differentiellen Variablen berechnet werden.

Für die Tests wurde in DAESOL die iterative Methode verwendet, DASPK arbeitete mit der simultanen Methode etwas besser als mit der iterativen. Da DDASAC pro BDF-Schritt eine Auswertung und Zerlegung der Jacobi-Matrix benötigt, ist das Verfahren zur Ableitungsgenerierung sehr aufwendig für dieses Beispiel (siehe Abbildung 7.12). Es benötigt in etwa den dreifachen Rechenaufwand wie DAESOL.

In DASPK wurde wiederum (wie für die reine Simulation auch) die Dense-Version verwendet. Zu Vergleichszwecken wurden die Berechnungen in DAESOL auch mit der Dense-Version durchgeführt, die Sparse-Version von DAESOL (▲) zeigt die für die Praxis relevanten Rechenzeiten (siehe Abbildung 7.12).

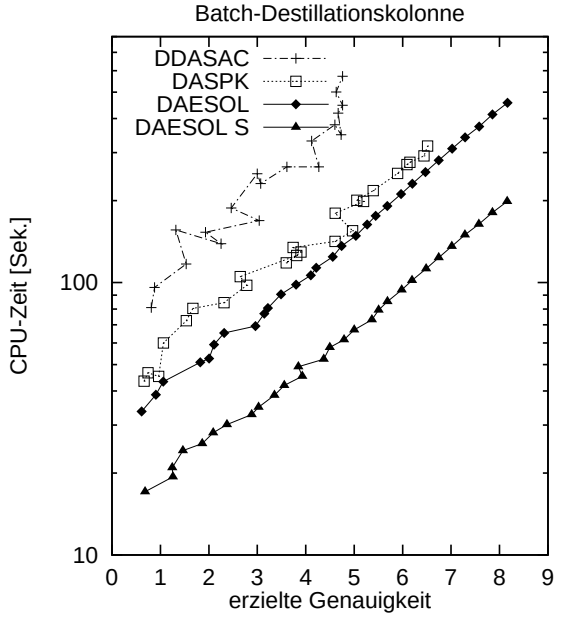


Abbildung 7.12: Benötigte CPU-Zeit der jeweiligen Integratoren in Abhängigkeit von der erreichten Genauigkeit für die Ableitungsgenerierung bei der Batch-Destillationskolonne.

Für die in Leineweber [Lei99] dargestellten reduzierten SQP-Verfahren sind an sich $n_y + n_q + 1 = 14$ *Richtungsableitungen in Richtung*

$$WD = \begin{pmatrix} I_{n_y} & 0_{n_y \times n_q} & 0 \\ WD_1^z & WD_2^z & WD_3^z \\ 0_{nq \times n_y} & I_{n_q} & 0 \end{pmatrix} \in I\!R^{n_y+n_q+1}$$

erforderlich. Die Matrizen $WD_1^z \in \mathbb{R}^{n_z \times n_y}$, $WD_2^z \in \mathbb{R}^{n_z \times n_q}$ und $WD_3^z \in \mathbb{R}^{n_z \times 1}$ sind dabei in der Regel voll besetzt. Der Integrator DASPK (wie auch DDASAC) stellt keine direkte Berechnung dieser Richtungsableitungen zur Verfügung. Hierfür müssen zunächst die Ableitungen nach allen Komponenten von x_0 und q (also $n_x + n_q = 225$) generiert und im Anschluß die Multiplikation $(W_{x_0}\, W_q) \cdot WD$ durchgeführt werden. In DAESOL können die Richtungsableitungen durch die speziellen Implementierungen direkt berechnet werden. Der Aufwand hängt nur von der Anzahl der Richtungen ab. Tabelle 7.6 zeigt den Vergleich von DASPK und DAESOL für die zu berechnenden Ableitungen in Richtung WD. Die Unterschiede in der CPU-Zeit liegen zum einen darin, daß DAESOL bei gleicher Anzahl von zu berechnenden Richtungen schon etwas schneller arbeitet als DASPK (siehe Abbildung 7.12), vor allem aber in der unterschiedlichen Anzahl der zu berechnenden Richtungen. Da DASPK für das Modell der Batch-Destillationskolonne mit der Dense-Version am schnellsten arbeitet, wurden für die Vergleiche die Rechnungen in DAESOL auch mit der Dense-Version durchgeführt. Die CPU-Zeiten für DAESOL liegen dabei drastisch unter denen von DASPK. Die Sparse-Version von DAESOL (DAESOL S) zeigt die für die Praxis relevanten Rechenzeiten, die nochmals deutlich geringer sind. Da für DASPK sehr viele Richtungen zu berechnen sind, war die direkte Methode zur Ableitungsgenerierung am schnellsten. Der Rechenaufwand bei DDASAC, das auch die direkte Methode zur Verfügung stellt, ist trotzdem im Durchschnitt nahezu doppelt so groß wie bei DASPK.

		# Schritte	CPU-Zeit [Sek.]
DDASAC	10^{-4}	335	$6.93 \cdot 10^2$
	10^{-6}	626	$1.22 \cdot 10^3$
	10^{-8}	—	—
DASPK	10^{-4}	242	$3.70 \cdot 10^2$
	10^{-6}	537	$8.16 \cdot 10^2$
	10^{-8}	1083	$1.65 \cdot 10^3$
DAESOL	10^{-4}	221	$6.78 \cdot 10^1$
	10^{-6}	425	$1.24 \cdot 10^2$
	10^{-8}	722	$2.02 \cdot 10^2$
DAESOL S	10^{-4}	221	$2.97 \cdot 10^1$
	10^{-6}	425	$5.08 \cdot 10^1$
	10^{-8}	721	$8.82 \cdot 10^1$

Tabelle 7.6: Aufwand für DDASAC, DASPK und DAESOL zur Generierung der Richtungsableitungen in Richtung *WD* für die Batch-Destillationskolonne. DAESOL S zeigt den Aufwand für die Sparse-Version von DAESOL.

Beispiel 7.9 (2D-Konvektions-Diffusionsgleichung (Fortsetzung))
Für das Modell der Konvektions-Diffusionsgleichung sollen die Ableitungen nach den beiden Parametern p_1 und p_2 berechnet werden. Da das DAE-System nach Diskretisierung

für den ODE- und DAE-Fall sehr groß ist, ist die direkte Methode wie in DDASAC wiederum viel zu aufwendig. Die Abbildungen 7.13 zeigen die CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die Konvektions-Diffusionsgleichung modelliert als ODE- bzw. DAE-System. Sowohl für DASPK als auch für DAESOL wurde die iterative Methode verwendet. DAESOL ist dabei im Durchschnitt etwas schneller als DASPK.

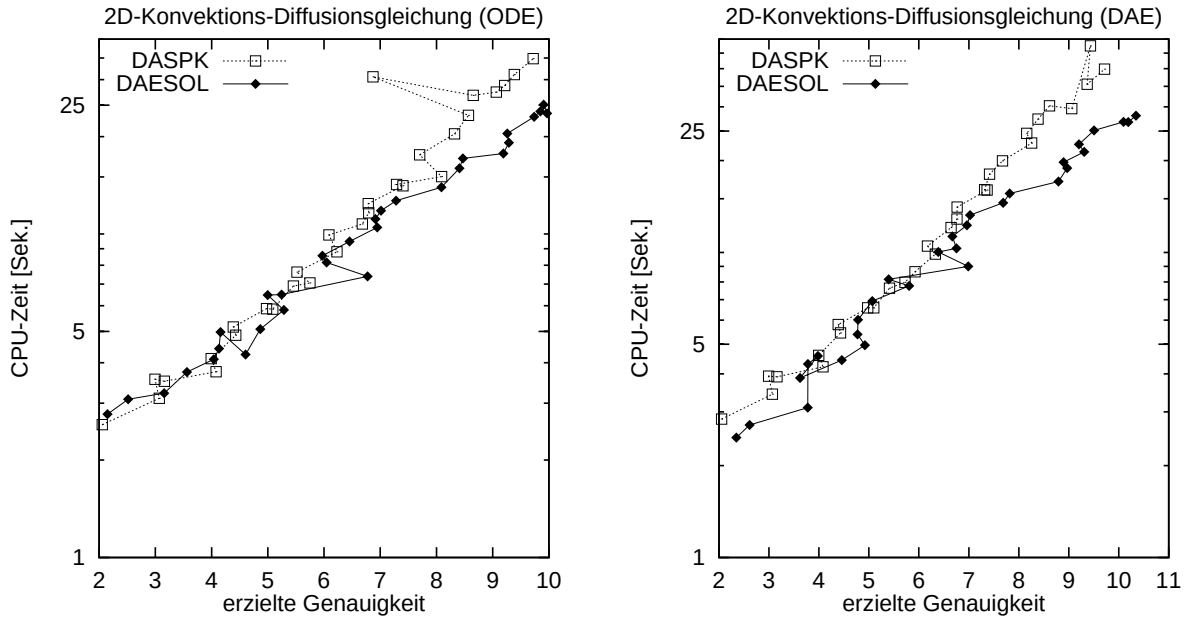


Abbildung 7.13: CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die Konvektions-Diffusionsgleichung modelliert als ODE- bzw. DAE-System.

Beispiel 7.10 (Urethan-Reaktion (Fortsetzung))
Für die Urethan-Reaktion sollen die Ableitungen nach den 8 *Parametern* E_{ai}, f_{ref_i}, $i = 1, 2, 4$, f_{c2} *und* dh_2 *und nach den Anfangswerten der differentiellen Variablen* $x_{i,0}$, $i = 3, \ldots, 5$, *berechnet werden.*

Zur Ableitungsgenerierung verwenden wir bei allen drei Integratoren (DDASAC, DASPK und DAESOL) die direkte Methode. Abbildung 7.14 zeigt die hierfür benötigte CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die zu berechnenden Ableitungen.

Für das Optimierungsproblem zur Versuchsplanung müssen für die Gradienten der Nebenbedingungen zusätzlich die Ableitungen nach den 56 Steuergrößen (7 Anfangsmolzahlen in Reaktor und Zulaufgefäßen, 46 Variablen aus der Parametrisierung der Steuerfunktionen und 3 Anfangswerte), für den Gradient des Zielfunktionals zusätzlich zweite gemischte Ableitungen nach Anfangswerten, Parametern und Steuergrößen berechnet werden. Tabelle 7.7 zeigt die CPU-Zeit für unterschiedliche Genauigkeiten für die reine Simulation (VDAE 0), die Generierung von ersten Ableitungen nach Anfangswerten und Parametern (VDAE p) und nach Anfangswerten und Steuergrößen (VDAE q) sowie die zweiten gemischten Ableitungen hierzu (VDAE pq). Bei der Berechnung der zweiten Ableitungen

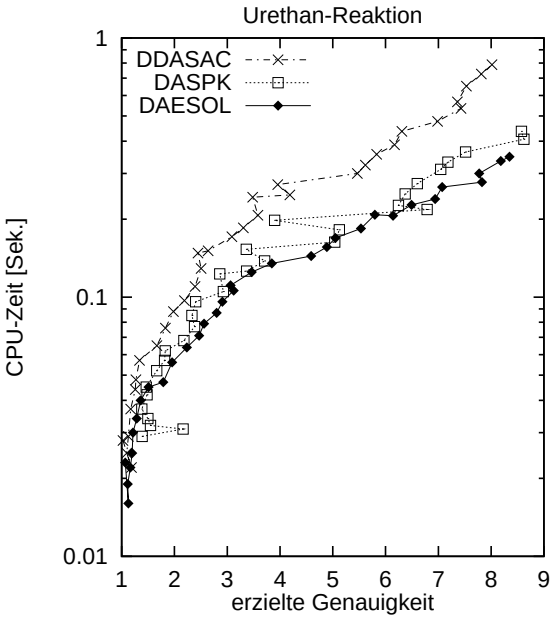


Abbildung 7.14: CPU-Zeit in Abhängigkeit von der erzielten Genauigkeit für die Berechnung 1. Ableitungen bei der Urethan-Reaktion.

werden die ersten Ableitungen (VDAE p) und (VDAE q) automatisch mitberechnet. Alle anderen Integratoren können keine zweiten Ableitungen berechnen.

TOL		VDAE 0	VDAE p	VDAE q	VDAE pq
CPU-Zeit	10^{-5}	0.03	0.09	0.29	3.68
	10^{-8}	0.05	0.19	0.82	9.88
	10^{-11}	0.09	0.43	1.64	22.1

Tabelle 7.7: CPU-Zeit für Simulation und Generierung von ersten und zweiten Ableitungen für die Urethan-Reaktion.

7.4 Anwendung auf eine sequentielle Vorgehensweise zur Parameterschätzung und Versuchsplanung

Im folgenden untersuchen wir zwei unterschiedliche Reaktionsmechanismen – die Phosphin- und Urethan-Reaktion – in Semi-Batch-Fahrweise. Die Modelle wurden zusammen mit Kud von der BASF AG (siehe auch Bauer et al. [BHK$^+$98, BKBS98]) aufgestellt. Das Ziel ist in beiden Fällen, die im Modell enthaltenen unbekannten Parameter möglichst gut (und mit möglichst wenigen Experimenten) zu schätzen. Wir verwenden hierfür eine sequentielle Vorgehensweise aus Versuchsplanung, Auswertung der

Experimente, Schätzung der Parameter und anschließender neuer Versuchsplanung unter Berücksichtigung der bereits gewonnenen Informationen. Für die numerische Lösung der Optimierungsprobleme zur Versuchsplanung werden unter anderem zweite gemischte Ableitungen der Lösungstrajektorie nach Anfangswerten, Parametern und Steuergrößen benötigt. Die Ergebnisse zur Versuchsplanung bei DAE-Systemen konnten somit erst mit den Neuentwicklungen in DAESOL erzielt werden.

Beispiel 7.11 (Phosphin-Reaktion)
Bei der Phosphin-Reaktion handelt es sich um die folgende chemische Reaktion in einem Semi-Batch-Reaktor

$$\underset{\text{Phosphin}}{A} + \underset{\text{Halogenid}}{B} \xrightarrow{k} \underset{\text{Phosphoniumsalz.}}{C}$$

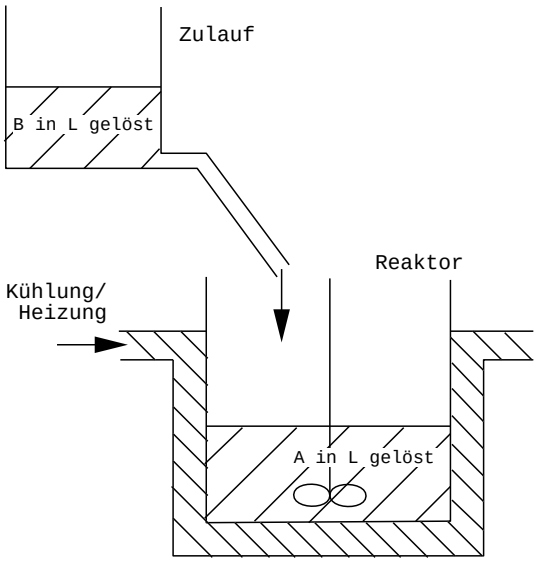


Abbildung 7.15: Reaktionsanordnung der Phosphin-Reaktion

Die Reaktion wurde im Hauptlabor der BASF AG (siehe Bauer et al. [BHK+98]) untersucht. Das Ziel war die Schätzung der nur ungenau bekannten kinetischen Parameter. Die Reaktionsgeschwindigkeit wird mit Hilfe des Arrhenius-Ansatzes modelliert

$$k(t) = f_{ref}\, e^{-\frac{E_a}{R}\left(\frac{1}{T(t)} - \frac{1}{T_{ref}}\right)}. \tag{7.7}$$

Da die Kinetikparameter, die Aktivierungsenergie E_a und der Frequenzfaktor f, häufig stark korelliert sind, wird eine Umparametrisierung – wie auch bei der Urethan-Reaktion – mit Hilfe der Referenztemperatur T_{ref} im Sinne von (7.7) vorgenommen. Dabei ist auch der Wert des Frequenzfaktors abhängig von der Referenztemperatur.

Unter Berücksichtigung des Massenwirkungsgesetzes und einer Differentialgleichung für die Wärmebilanz führt die Modellierung der Phosphin-Reaktion auf das folgende nichtlineare DAE-System:

$$\begin{aligned}
\dot{n}_A(t) &= -V(t)^{1-\alpha_1-\alpha_2}\, k(t)\, n_A(t)^{\alpha_1}\, n_B(t)^{\alpha_2} \\
n_B(t) &= -n_C(t) + n_{B,in} \cdot inflow(t) \\
n_C(t) &= -n_A(t) + n_A(0) \\
n_L(t) &= n_L(0) + n_{L,in} \cdot inflow(t) \\
\dot{T}(t) &= \underbrace{(dH\, V(t)^{1-\alpha_1-\alpha_2}\, k(t)\, n_A(t)^{\alpha_1}\, n_B(t)^{\alpha_2}}_{Reaktion} \\
&+ \underbrace{\dot{m}(t)\, c_p\, (Tp_0 - T(t))}_{Zulauf} \\
&+ \underbrace{kd\, A_W(t)\, (T_W(T_{in}(t)) - T(t))}_{Kühlung})/(c_p\, m(t))
\end{aligned}$$

mit der Gesamtmasse im Reaktor

$$m(t) = \sum_{i \in \{A,B,C,L\}} M_i\, n_i(t),$$

dem Volumen im Reaktor

$$V(t) = m(t)/\rho,$$

der Höhe des Volumens im zylinderförmigen Reaktor

$$h(t) = 4\, V(t)/(1000\, d^2\, \pi),$$

der Wärmeaustauschfläche

$$A_W(t) = \pi\, d\, h(t) + \pi\, d^2/4.0$$

und der mittleren Kühlwassertemperatur

$$T_W(t) = \frac{kd\, A_W(t)\, T(t) + 2.0\, V_W\, \rho_W\, c_{WW}\, T_{in}(t)}{kd\, A_W(t) + 2.0\, V_W\, \rho_W\, c_{WW}}.$$

Verfahrenstechnische Komponenten im Modell sind der Zulauf $inflow(t)$ und die Temperatur $T_{in}(t)$ der Heizung bzw. Kühlung, außerdem die Anfangsmolzahlen von Phosphin und dem Lösungsmittel im Reaktor $n_A(0)$ und $n_L(0)$ und die Anfangstemperatur des Gemischs $T(0)$.

Die zusätzlich im Modell auftretenden Konstanten haben die folgende Bedeutung und Wert:

dH	108 000	*molare Reaktionsgeschwindigkeit*
c_p	2.0	*spezifische Wärmekapazität*
Tp_0	293.16	*Temperatur im Zulaufgefäß (konstant auf Raumtemperatur gesetzt)*
kd	1 600	*Wärmedurchgangskoeffizient*
d	0.15	*Durchmesser des zylinderförmigen Reaktors*
V_W	1.0	*Volumenstrom des Wärmeträgers*
ρ_W	900	*Dichte des Wärmeträgers*
c_{WW}	0.12	*Wärmekapazität des Wärmeträgers*
ρ	1 000	*Dichte der Reaktionslösung*
M_A	261.32	*Molmasse von Phosphin (A)*
M_B	122.60	*Molmasse von Halogenid (B)*
M_C	383.92	*Molmasse des Phosphoniumsalzes (C)*
M_L	100.00	*Molmasse des Lösungsmittels (L)*
T_{ref}	365.16	*Referenztemperatur.*

Neben den Kinetikparametern E_a und f_{ref} sollen auch die Reaktionsordnungen α_1 und α_2 bestimmt werden.

Als Messungen stehen die Molzahl der Spezies C (n_3) mit einem normalverteilten Meßfehler mit Standardabweichung 0.01 und die Temperatur T mit einem normalverteilten Meßfehler mit Standardabweichung 0.5 zur Verfügung.

Jedes Experiment dauert 180 Minuten, die Messungen können alle 12 Minuten durchgeführt werden. Maximal dürfen pro Experiment 5 Proben von Spezies C und 5 Temperaturmessungen durchgeführt werden.

Wir parametrisieren die Steuerfunktionen durch stückweise lineare und stetige Funktionen mit 8 Parametrisierungsintervallen für den Zulauf und 6 Parametrisierungsintervallen für die Temperatursteuerung. Der Zulauf gibt das Integral über die bereits zugeflossene Menge (normiert auf 1) an; nach 90 Minuten soll bereits alles zugeflossen sein, d. h.

$$inflow(0) = 0,\ inflow(t) = 1,\ t \in [90; 180].$$

Die Zulaufrate muß

$$\frac{\mathrm{d}}{\mathrm{d}t} inflow(t) \geq 0,\ t \in [0; 90].$$

erfüllen, für die Temperatursteuerung der Heizung bzw. Kühlung gelte

$$273.16 \leq T_{in}(t) \leq 393.16,\ t \in [0; 180],$$

dabei darf die Temperaturänderung pro Minute maximal 2 $[K]$ betragen:

$$-2 \leq \frac{\mathrm{d}}{\mathrm{d}t} T_{in}(t) \leq 2,\ t \in [0; 180].$$

Die Steuergrößen können innerhalb der folgenden Grenzen gewählt werden

$$\begin{aligned} n_A(0) &\in [0.8; 20] \\ n_L(0) &\in [3; 14] \\ T(0) &\in [323.16; 351.16]. \end{aligned}$$

Wir erhalten so pro Experiment 3 Steuergrößen, 28 Variablen aus der Parametrisierung der Steuerfunktionen und 30 Gewichte an die möglichen Messungen, also 61 Versuchsplanungsvariablen.

Erste Schätzungen für die Aktivierungsenergie und den Frequenzfaktor wurden aus Literaturdaten aus ähnlichen Reaktionen übernommen (siehe z. B. Bauer et al. [BHK+98]), die Reaktionsordnungen wurden zunächst auf 1 *gesetzt.*

Wir berechnen einen Versuchsplan für ein Experiment mit dem D-Gütekriterium. Für dieses Experiment werden die benötigten Meßdaten erhoben. Da zunächst keine Experimente im Labor durchgeführt werden konnten, wurde der Prozeß für die „wahren“ Parameterwerte simuliert. Zur Erzeugung von (Pseudo-)Meßdaten wurden die Modellantworten mit einem normalverteilten Fehler in der Größe des jeweiligen Meßfehlers überlagert.

Exp.	α_1	α_2	E_a ($\cdot 8 \cdot 10^4$)	f_{ref}
Startw.	1	1	1	2.14
1	5 ± 16	4 ± 14	2 ± 2	0.5 ± 1
2	0.81 ± 0.07	1.17 ± 0.1	1.03 ± 0.02	0.40 ± 0.02
3	0.799$\pm$0.005	1.196$\pm$0.005	1.0345$\pm$0.0035	0.4025$\pm$0.0045
4	0.800$\pm$0.003	1.197$\pm$0.005	1.035 ± 0.003	0.403 ± 0.004
5	0.800$\pm$0.003	1.199$\pm$0.001	1.0362$\pm$0.0008	0.405 ± 0.001

Tabelle 7.8: Sukzessive Parameterschätzung und Versuchsplanung für die Phosphin-Reaktion. Die Tabelle zeigt die Startwerte für die Parameter sowie die Schätzungen nach jeweils einem weiteren Experiment mit der zugehörigen (näherungsweisen) Standardabweichung.

Aus den Meßdaten für das erste Experiment werden die Parameter geschätzt. Da die zugehörigen (näherungsweisen) Standardabweichungen noch sehr groß sind (siehe Tabelle 7.8), wurde ein weiteres Experiment ebenfalls mit dem D-Kriterium als Zielfunktional geplant. Die bereits aus den Meßdaten des ersten Experiments gewonnene Information wird bei der Optimierung des zweiten Experiments berücksichtigt. Dieser Vorgang wird solange wiederholt, bis die Standardabweichungen der geschätzten Parameter als klein genug akzeptiert werden. Die Abbildungen 7.16 bis 7.20 zeigen den Lösungsverlauf der Spezies $n_1, \ldots, n_3$ und der Temperatur T im Reaktor, das Dosierprofil $inflow$ und das Temperaturprofil T_{in} der Heizung bzw. Kühlung für die jeweiligen Experimente.

Tabelle 7.9 zeigt die vom Optimierungsproblem gewählten Werte für die Steuergrößen.

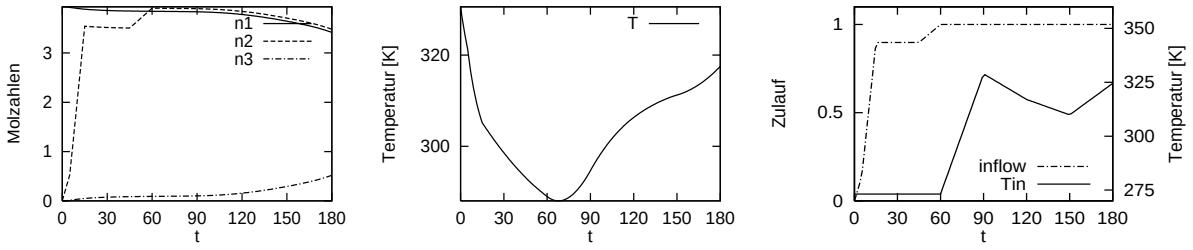


Abbildung 7.16: Erstes Experiment zur Phosphin-Reaktion.

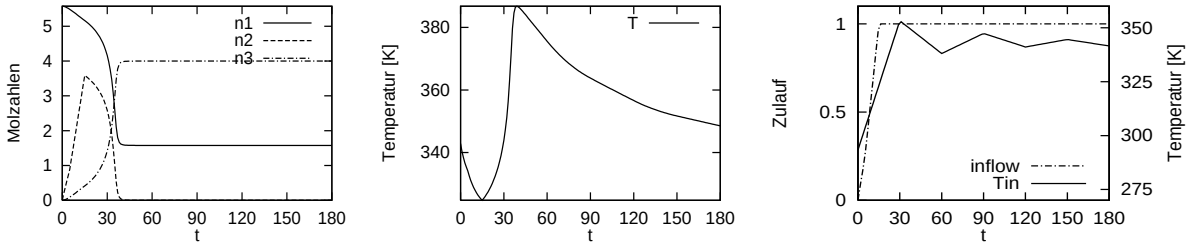


Abbildung 7.17: Zweites Experiment zur Phosphin-Reaktion.

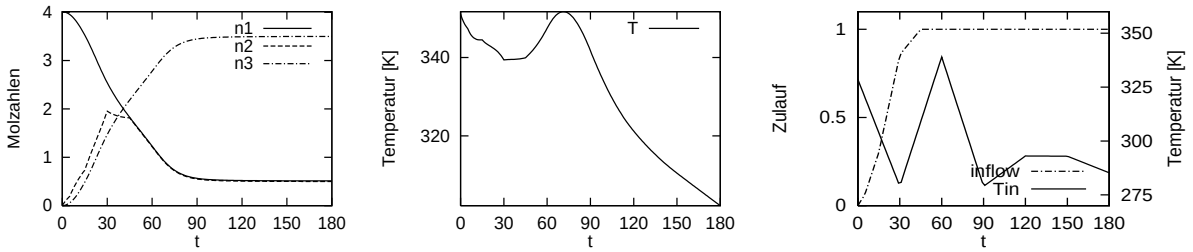


Abbildung 7.18: Drittes Experiment zur Phosphin-Reaktion.

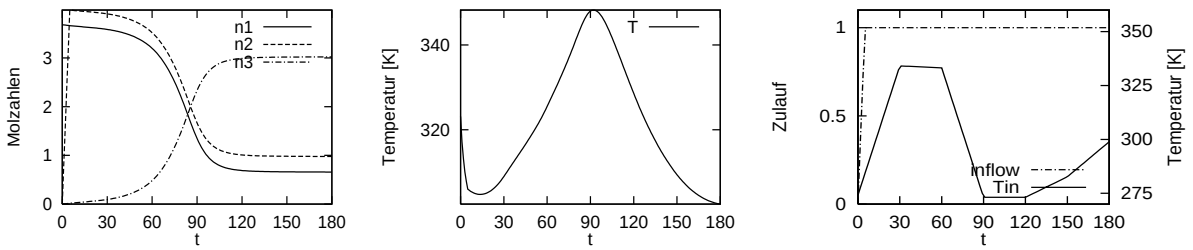


Abbildung 7.19: Viertes Experiment zur Phosphin-Reaktion.

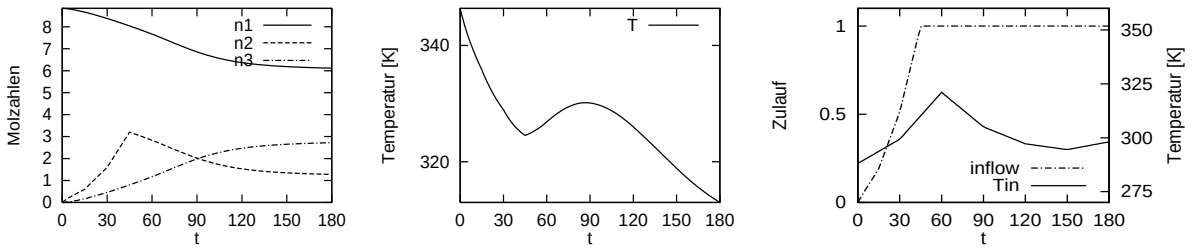


Abbildung 7.20: Fünftes Experiment zur Phosphin-Reaktion.

Experiment	$n_A(0)$	$n_L(0)$	$T(0)$
1	3.94	3.0	330.59
2	5.58	3.0	342.85
3	4.01	3.0	351.16
4	3.68	3.0	323.16
5	8.84	14.0	346.39

Tabelle 7.9: Optimierte Werte für die Steuergrößen für die Phosphin-Reaktion.

Experiment	Messungen C	Messungen T
1	36.0, 144.0, 156.0, 168.0, 180.0	12.0, 24.0, 36.0, 48.0, 60.0
2	12.0, 24.0, 36.0, 168.0, 180.0	12.0, 24.0, 36.0, 48.0, 60.0
3	36.0, 48.0, 60.0, 72.0, 84.0	36.0, 48.0, 72.0, 84.0, 96.0
4	84.0, 96.0, 108.0, 120.0, 132.0	72.0, 84.0, 96.0, 108.0, 120.0
5	132.0, 144.0, 156.0, 168.0, 180.0	72.0, 84.0, 96.0, 108.0, 120.0

Tabelle 7.10: Auswahl der Messungen (gerundet) für die Phosphin-Reaktion.

Die Gewichte an die Messungen für die optimierten Versuchspläne waren nahezu 0/1 und sind gerundet wie in Tabelle 7.10 angegeben.

Die näherungsweise Kovarianzmatrix hat nach fünf Experimenten die Einträge

$$C = \begin{pmatrix} 9.33 \cdot 10^{-6} & 3.21 \cdot 10^{-6} & -3.95 \cdot 10^{-7} & -1.14 \cdot 10^{-6} \\ 3.21 \cdot 10^{-6} & 2.68 \cdot 10^{-6} & 6.65 \cdot 10^{-7} & 7.43 \cdot 10^{-7} \\ -3.95 \cdot 10^{-7} & 6.65 \cdot 10^{-7} & 6.06 \cdot 10^{-7} & 8.51 \cdot 10^{-7} \\ -1.14 \cdot 10^{-6} & 7.43 \cdot 10^{-7} & 8.51 \cdot 10^{-7} & 1.25 \cdot 10^{-6} \end{pmatrix}.$$

Nach drei Experimenten ist die nähernungsweise Standardabweichung für die geschätzten Parameter bereits bei einem Prozent, nach fünf Experimenten für alle Parameter kleiner als drei Promille.

Bemerkung 7.4.1 (Intuitiver Versuchsplan zur Phosphin-Reaktion)
Von einem erfahrenen Chemiker der BASF AG wurde ein inutitiver Versuchsplan bestehend aus fünf Experimenten aufgestellt (siehe Bauer et al. [BHK+98]). Die Rahmenbedingungen waren dabei die gleichen wie für die methodengestützten Versuchspläne.

Nach fünf Experimenten erhält man folgende Schätzungen für die Parameter mit den folgenden näherungsweisen Standardabweichungen

$$\begin{aligned} \alpha_1 &= 0.79 \pm 0.02 \\ \alpha_2 &= 1.19 \pm 0.01 \\ E_a &= 1.037 \pm 0.005 \\ f &= 0.406 \pm 0.006. \end{aligned}$$

Die Standardabweichungen der geschätzten Parameter sind nach fünf Experimenten etwa zwei Prozent, also immer noch größer als bei den methodengestützten Versuchsplänen nach drei Experimenten.

Beispiel 7.12 (Urethan-Reaktion (Fortsetzung))
Für das in Abschnitt 7.2 dargestellte Modell der Urethan-Reaktion sollen die 8 unbekannten Parameter – die Frequenzfaktoren f_{ref_i}, $i = 1, 2, 4$, die Aktivierungsenergien E_{ai}, $i = 1, 2, 4$, sowie der Vorfaktor f_{c2} und die Reaktionsenthalpie dh_2 der Rückreaktion – geschätzt werden.

Der Experimentator stellt dabei normalerweise nicht die in Abschnitt 7.2, Beispiel 7.5 angebenen Steuergrößen – Anfangszusammensetzung im Reaktor $n_1(0)$, $n_2(0)$ und $n_6(0)$ und in den beiden Zulaufgefäßen $n_{1ea,0}$ und $n_{6ea,0}$ beziehungsweise $n_{2eb,0}$ und $n_{6eb,0}$ – ein, sondern die Molverhältnisse MV_i, die Wirkstoffgehaltangaben in Reaktor g_a und Zulaufgefäßen g_{aea} und g_{aeb} und das Anfangsvolumen V_a. Zwischen den vom Experimentator einstellbaren Größen und den Anfangsmolzahlen besteht der folgende Zusammenhang:

$$\begin{aligned} MV_1 &= \frac{n_2(0) + n_{2eb,0}}{n_1(0) + n_{1ea,0}} \\ MV_2 &= \frac{n_{1ea,0}}{n_1(0)} \end{aligned}$$

$$
\begin{aligned}
MV_3 &= \frac{n_{2eb,0}}{n_1(0)} \\
g_a &= \frac{n_1(0)\cdot M_1 + n_2(0)\cdot M_2}{n_1(0)\cdot M_1 + n_2(0)\cdot M_2 + n_6(0)\cdot M_6} \\
g_{aea} &= \frac{n_{1ea,0}\cdot M_1}{n_{1ea,0}\cdot M_1 + n_{6ea,0}\cdot M_6} \\
g_{aeb} &= \frac{n_{2eb,0}\cdot M_2}{n_{2eb,0}\cdot M_2 + n_{6eb,0}\cdot M_6} \\
V_a &= \frac{n_1(0)}{\rho_1}\cdot M_1 + \frac{n_2(0)}{\rho_2}\cdot M_2 + \frac{n_6(0)}{\rho_6}\cdot M_6
\end{aligned}
$$

Aus den Anfangsmolzahlen erhält man die einzuwiegenden Massen für die Reagenzien NCO, BUOH und DMSO im Reaktor (m_{NCO}, m_{BUOH}, m_{DMSO}) bzw. in Zulauf a ($m_{NCO,ea}$, $m_{DMSO,ea}$) und Zulauf b ($m_{BUOH,eb}$, $m_{DMSO,eb}$), wie in den folgenden Gleichungen modelliert:

$$
\begin{aligned}
M_1\cdot n_1(0) &= 0.9\cdot m_{NCO} \\
M_2\cdot n_2(0) &= 1.0\cdot m_{BUOH} \\
M_6\cdot n_6(0) &= 0.1\cdot m_{NCO} + 1.0\cdot m_{DMSO} \\
M_1\cdot n_{1ea,0} &= 0.9\cdot m_{NCO,ea} \\
M_6\cdot n_{6ea,0} &= 0.1\cdot m_{NCO,ea} + 1.0\cdot m_{DMSO,ea} \\
M_2\cdot n_{2eb,0} &= 1.0\cdot m_{BUOH,eb} \\
M_6\cdot n_{6eb,0} &= 1.0\cdot m_{DMSO,eb}
\end{aligned}
$$

Als Versuchsplanungsgrößen können die Molverhältnisse MV_1, MV_2, MV_3, der Wirkstoffgehalt im Reaktor g_a und in Zulauf a g_{aea} und Zulauf b g_{aeb} sowie das Anfangsvolumen V_a innerhalb folgender Grenzen gewählt werden:

$$
\begin{aligned}
MV_1 &\in [0.1; 10] \\
MV_2 &\in [0; 1000] \\
MV_3 &\in [0; 10] \\
g_a &\in [0; 0.8] \\
g_{aea} &\in [0; 0.9] \\
g_{aeb} &\in [0; 1] \\
V_a &\in [0; 0.00075].
\end{aligned}
$$

Ein Experiment dauert maximal 80 Stunden. Wir parametrisieren die kontinuierlichen Steuerfunktionen für die Temperatur und die Zuläufe, indem wir sie durch jeweils sieben stückweise lineare und stetige Polynome ersetzen. Die Zulaufprofile beschreiben – wie bei der Phosphin-Reaktion auch – das Integral über die bereits zugeflossene Menge (normiert auf 1), d. h. $inflow_\kappa(0) = 0$, $inflow_\kappa(80) = 1$, $\kappa \in \{a, b\}$.

Für die Temperatur im Reaktor gelte

$$273.16 \leq T(t) \leq 473.16$$

Nur tagsüber darf Zulauf stattfinden und die Temperatur geändert werden, deshalb müssen nachts die Zulaufraten und die Heiz-/Kühlrate der Innentemperatur gleich 0 sein:

$$\left.\begin{array}{l} \frac{\mathrm{d}}{\mathrm{d}t} inflow_\kappa(t) \geq 0 \\ -40 \leq \frac{\mathrm{d}}{\mathrm{d}t} T(t) \leq 40 \end{array}\right\} \quad \text{für } t \in [0;8],\ [24;32],\ [8;56],\ [72;80]$$

$$\frac{\mathrm{d}}{\mathrm{d}t} inflow_\kappa(t) = \frac{\mathrm{d}}{\mathrm{d}t} T(t) = 0 \quad \text{sonst.}$$

Zur Ermittlung von Meßdaten stehen drei Meßverfahren zur Auswahl:

- *Titration: Massenprozent von A (Standardabweichung des Meßfehlers:* 0.5*)*
- *HPLC, Säule 1: Massenprozent von C (Standardabweichung des Meßfehlers:* 0.5*) und von D (Standardabweichung des Meßfehlers:* 0.005*)*
- *HPLC, Säule 2: Massenprozent von E (Standardabweichung des Meßfehlers:* 0.0005*).*

Es wurden folgende 10 mögliche Meßzeitpunkte gewählt: nach 0.5, 1, 4, 8, 24, 32, 46, 56, 72 und 80 Stunden. An jedem Zeitpunkt kann jedes der drei Meßverfahren eingesetzt werden. Pro Experiment sollen von den insgesamt 30 möglichen Messungen maximal 16 durchgeführt werden.

Insgesamt erhalten wir bei der Optimierung pro Experiment 97 Versuchsplanungsvariablen: 7 Steuergrößen, 7 Anfangsmolzahlen, 7 Reagenzien, 30 Meßgewichte und 46 Variablen aus der Parametrisierung der Steuerfunktionen.

Wir planen sukzessive drei Experimente. Als Anfangswerte verwenden wir dabei jeweils ein Experiment mit den in Abbildung 7.21 angegebenen Steuerfunktionen und Lösungstrajektorien. Tabelle 7.11 zeigt die geschätzten Parameter und die zugehörigen näherungsweisen Standardabweichungen. Die Parameter wurden dabei zu Beginn der sequentiellen Vorgehensweise auf 1 skaliert.

Die Abbildungen 7.22 bis 7.24 zeigen die Dosierprofile, die Temperatursteuerung und die Trajektorien der Spezies A bis E. Die Steuergrößen wurden wie in Tabelle 7.12 angegeben gewählt.

Die Gewichte für die Messungen der optimierten Versuchspläne waren nahezu 0/1*. Nach Rundung ergaben sich die folgenden Auswertezeitpunkte für die verschiedenen Meßverfahren:*

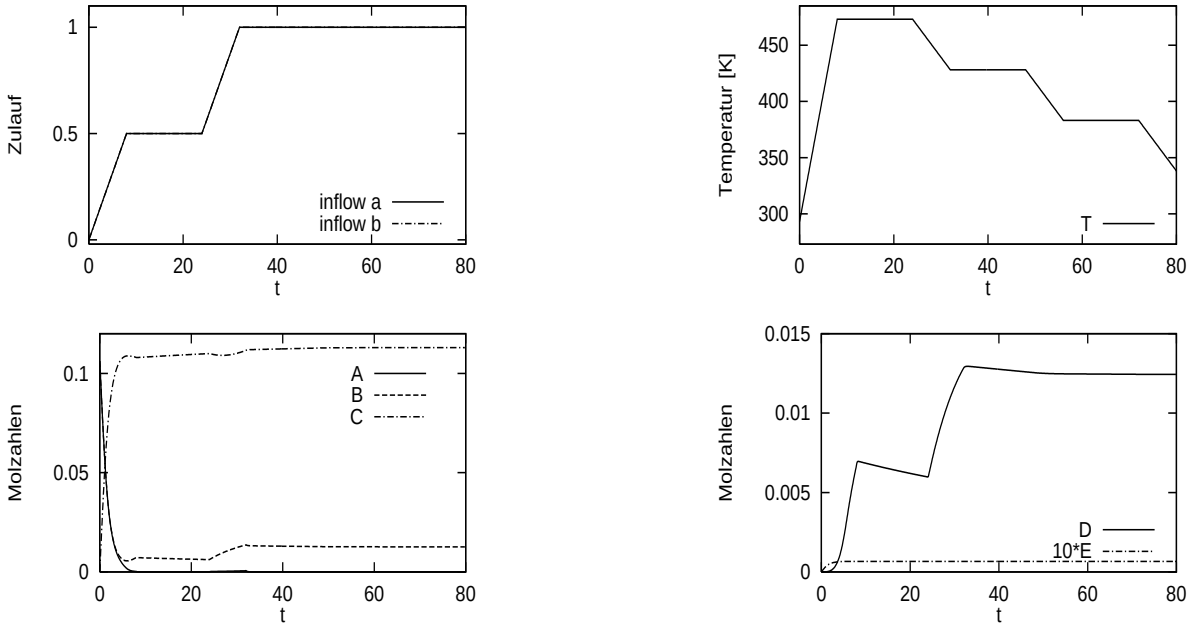


Abbildung 7.21: Starttrajektorien für die Steuerfunktionen und zugehörige Lösungsverläufe für die Urethan-Reaktion.

	Startp.	1. Experiment	2. Experiment	3. Experiment
k_{ref_1}	1	$2.57 \quad \pm 3 \cdot 10^{-2}$	$2.504 \quad \pm 6 \cdot 10^{-3}$	$2.506 \quad \pm 5 \cdot 10^{-3}$
E_{a1}	1	$0.826 \quad \pm 5 \cdot 10^{-3}$	$0.835 \quad \pm 1 \cdot 10^{-3}$	$0.835 \quad \pm 1 \cdot 10^{-3}$
k_{ref_2}	1	$91.1 \quad \pm 2 \cdot 10^{-1}$	$91.23 \quad \pm 1 \cdot 10^{-2}$	$91.231 \quad \pm 9 \cdot 10^{-3}$
E_{a2}	1	$0.8358 \pm 3 \cdot 10^{-4}$	$0.83546 \pm 9 \cdot 10^{-5}$	$0.83545 \pm 7 \cdot 10^{-5}$
k_{ref_4}	1	$58.3 \quad \pm 1 \cdot 10^{-1}$	$57.996 \quad \pm 4 \cdot 10^{-3}$	$57.000 \quad \pm 1 \cdot 10^{-3}$
E_{a4}	1	$0.655 \quad \pm 1 \cdot 10^{-3}$	$0.6577 \quad \pm 3 \cdot 10^{-4}$	$0.6572 \quad \pm 1 \cdot 10^{-4}$
dh_2	1	$1.085 \quad \pm 9 \cdot 10^{-3}$	$1.075 \quad \pm 9 \cdot 10^{-3}$	$1.083 \quad \pm 3 \cdot 10^{-3}$
k_{c2}	1	$1.29 \quad \pm 2 \cdot 10^{-2}$	$1.28 \quad \pm 1 \cdot 10^{-2}$	$1.288 \quad \pm 5 \cdot 10^{-3}$

Tabelle 7.11: Sukzessive Parameterschätzung und Versuchsplanung für die Urethan-Reaktion. Die Tabelle zeigt die Startwerte für die Parameter sowie die Schätzungen nach jeweils einem weiteren Experiment mit der zugehörigen (näherungsweisen) Standardabweichung.

1. Experiment:

- *von A: nach 0.5, 1 und 4 Stunden*
- *von C und D: nach 0.5, 1, 4, 8, 24, 32, 48, 56, 72 und 80 Stunden*
- *von E: nach 4, 8 und 56 Stunden*

2. Experiment:

- *von A: keine*
- *von C und D: nach 1, 8, 24, 32, 48, 56, 72 und 80 Stunden*
- *von E: nach 4, 8, 24, 32, 48, 56, 72 und 80 Stunden*

3. Experiment:

- *von A: keine*
- *von C und D: nach 24, 32, 48, 56, 72 und 80 Stunden*
- *von E: nach 0.5, 1, 4, 8, 24, 32, 48, 56, 72 und 80 Stunden*

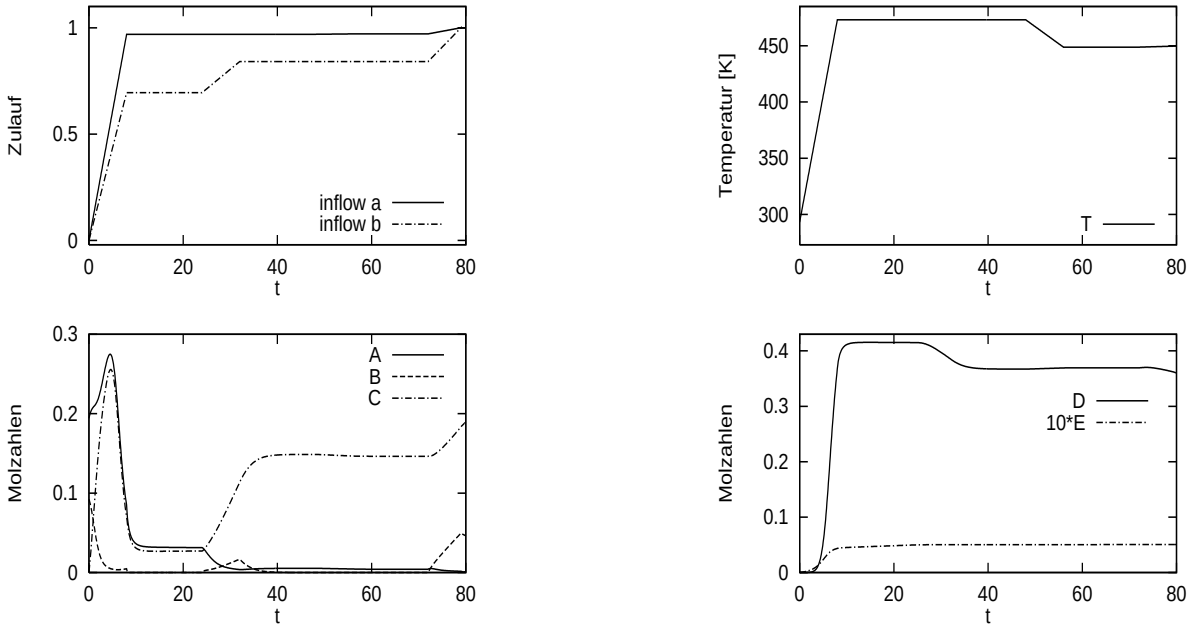


Abbildung 7.22: Erstes Experiment zur Urethan-Reaktion.

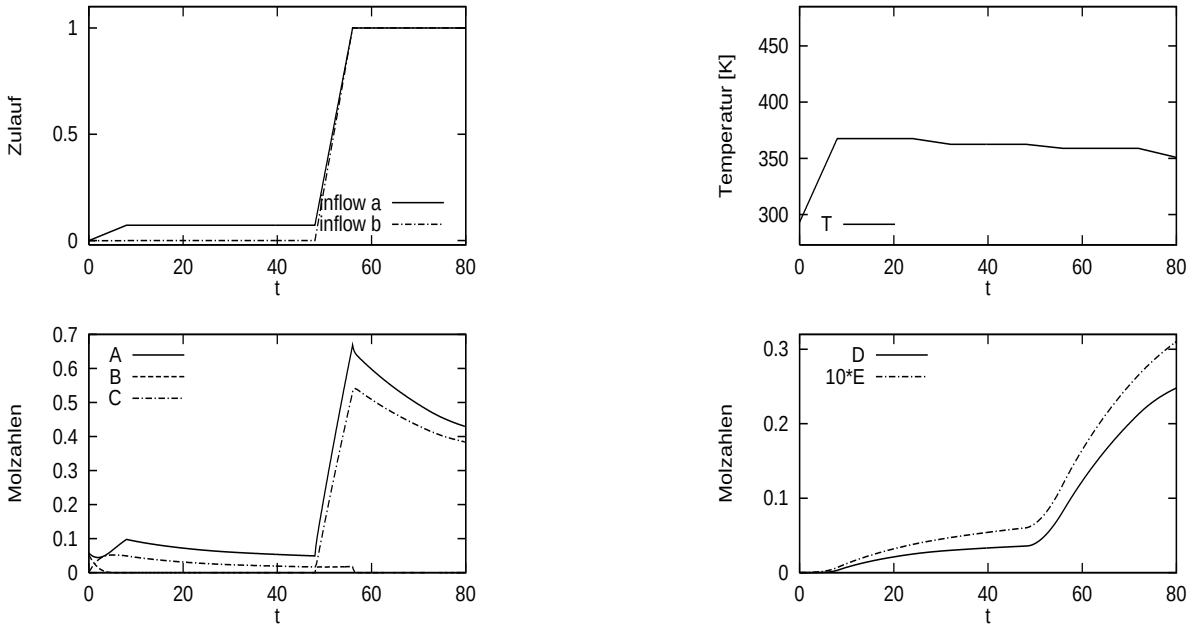


Abbildung 7.23: Zweites Experiment zur Urethan-Reaktion.

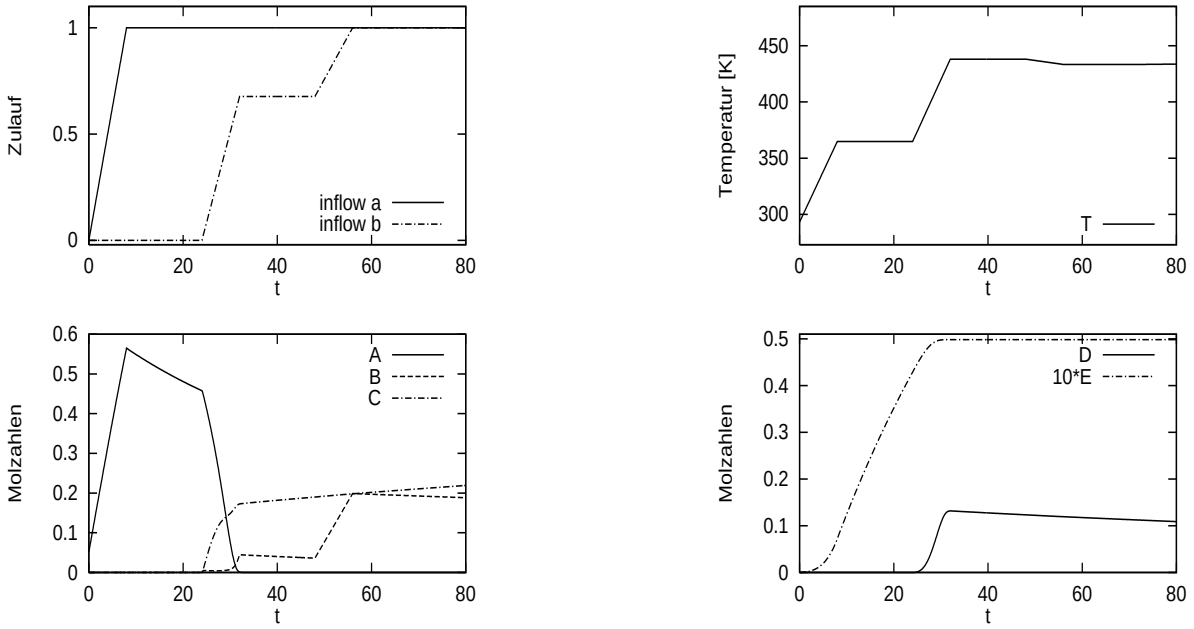


Abbildung 7.24: Drittes Experiment zur Urethan-Reaktion.

	1. Experiment	2. Experiment	3. Experiment
MV_1	0.644	0.451	0.880
MV_2	3.767	23.2	10.36
MV_3	2.593	10.0	10.0
g_a	0.8	0.8	0.8
g_{aea}	0.9	0.9	0.9
g_{aeb}	1.0	1.0	1.0
V_a	$3.64 \cdot 10^{-5}$	$1.36 \cdot 10^{-5}$	$7.01 \cdot 10^{-6}$

Tabelle 7.12: Optimierte Werte für die Steuergrößen für die Urethan-Reaktion.

Die methodengestützte Versuchsplanung bei der Phosphin-Reaktion zeigt gegenüber dem intuitiv aufgestellten Versuchsplan ein drastisches Einsparpotential. Ähnliches zeigte sich auch für die Urethan-Reaktion (siehe auch Bauer et al. [BHK$^+$98, BKBS98]). Die numerische Lösung des Versuchsplanungs-Optimierungsproblems war erst mit den Neuentwicklungen in DAESOL möglich. Durch die Kombination aus semi-analytischen Ableitungen, Automatischer Differentiation und Interner Numerischer Differentiation können die für die Optimierung benötigten Gradienten mit der vom Optimierungsverfahren geforderten Genauigkeit berechnet werden.

Mittlerweile wurden auch schon erste Ergebnisse zur Versuchsplanung für Parameterschätzprobleme bei nichtlinearen Transport- und Abbauprozessen von Xenobiotica (siehe Dieses et al. [DSBR99]) erzielt. Die Modelle enthalten – nach Ortsdiskretisierung – mehrere hundert Zustandsvariablen.

Kapitel 8

Zusammenfassung und Ausblick

Die vorliegende Arbeit dokumentiert die Leistungsfähigkeit der entwickelten Verfahren zur Simulation, Parameterschätzung und Versuchsplanung in DAE-Systemen. Wir wollen die wichtigsten Punkte hierbei zusammenfassen und offene Fragen wie auch weiterführende Arbeiten diskutieren.

8.1 Zusammenfassung der Arbeit

Wir haben ein BDF-Verfahren vorgestellt (Kapitel 2), das die allgemeine Klasse von Anfangswertproblemen bei DAEs vom Index 1 in linear impliziter Form löst, wie sie sehr häufig bei der Modellierung von Prozessen in Chemie und Verfahrenstechnik auftreten. Insbesondere wurde dabei auf eine effiziente Behandlung der Startphase – konsistente Initialisierung und Runge-Kutta-Starter – Wert gelegt (Kapitel 3) und auf die Generierung der im Optimierungskontext benötigten ersten und zweiten Ableitungen der Lösung des DAE-Systems (Kapitel 6). Aufbauend auf dem Optimierungsproblem zur Parameterschätzung wurde die Lösung des Optimal-Steuerungsproblems zur Versuchsplanung (Kapitel 5) dargestellt.

Die Methoden wurden im Programmpaket DAESOL implementiert. DAESOL enthält zur Lösung der Anfangswertprobleme und zur Ableitungsgenerierung die folgenden Techniken:

- Das BDF-Verfahren ist von variabler Ordnung und Schrittweite. Die Formeln zur Fehlerschätzung und Schrittweitensteuerung basieren auf dem tatsächlichen nichtäquidistanten Gitter.

- Eine Monitor-Strategie für das vereinfachte Newton-Verfahren reduziert den Aufwand zur Lösung der nichtlinearen Gleichungssysteme, insbesondere die Anzahl der Auswertungen der Ableitungen der Modellfunktionen des DAE-Systems. Diese werden adaptiv nach dem Kontraktionssatz von Bock [Boc87] kontrolliert.

- Ein Homotopie-Verfahren berechnet konsistente Anfangswerte bei stark nichtlinearen algebraischen Gleichungen. Es enthält eine Schrittweitensteuerung und eine Monitor-Strategie (ähnlich wie für die Nominaltrajektorie) für das vereinfachte Newton-Verfahren zur Lösung der Nullstellenprobleme, die beide adaptiv nach dem Kontraktionssatz für Homotopie-Verfahren bei Parameterschätzproblemen von Bock [Boc87] kontrolliert werden.

- Die relaxierte Formulierung der algebraischen Gleichungen mit Dämpfungsfaktor erlaubt – gerade im Optimierungskontext – die Integration mit inkonsistenten Anfangswerten. Zudem wird die Lösung im Verlauf der Integration an die Mannigfaltigkeit der ursprünglichen algebraischen Gleichungen angenähert, was die numerische Lösung der Optimierungsprobleme in der Regel robuster macht.

- Ein speziell (im Hinblick auf Fehlerschätzung und Lösung der Lineare-Algebra-Teilprobleme) auf das BDF-Verfahren zugeschnittenes SDIRK-Verfahren reduziert den Aufwand in der Startphase.

- Das Verfahren verfügt zudem über eine effiziente Generierung der für die Optimierung benötigten ersten und zweiten gemischten Ableitungen. Dabei werden unterschiedliche Varianten – direkte und iterative Methode – bereitgestellt, die den unterschiedlichen Anforderungen (je nach Größe des Systems und Anzahl der zu berechnenden Ableitungen) Rechnung tragen.

- DAESOL gestattet die direkte Berechnung von Richtungsableitungen, wie sie im reduzierten Ansatz zur Parameterschätzung (siehe auch Schlöder [Sch88]) – und somit auch zur Versuchsplanung – und für das von Leineweber [Lei99] entwickelte Verfahren benötigt werden.

Mit Hilfe dieser Techniken und insbesondere der effizienten Generierung der ersten und zweiten Ableitungen der Lösung des DAE-Systems konnten erstmals die Optimal-Steuerungsprobleme zur Versuchsplanung bei chemischen Reaktionssystemen erfolgreich gelöst werden. Diese verfügen über die folgenden Strategien und Eigenschaften:

- direkter Ansatz zur Lösung des Optimal-Steuerungsproblems;

- relaxierte Formulierung der 0/1-Bedingungen an die Gewichte der Messungen;

- das zugrundeliegende Parameterschätzproblem für die Versuchsplanung kann beschränkt sein;

- die allgemeine Formulierung erlaubt die Maximierung nicht nur der statistischen Güte der zu schätzenden Parameter, sondern auch von abgeleiteten Größen;

- Ausnutzen der Mehrfachexperimentstruktur, insbesondere Berücksichtigung der Information aus bereits durchgeführten Experimenten.

Dabei wurde insbesondere auf eine effiziente Berechnung der für die numerische Lösung der resultierenden Optimierungsprobleme mit einem SQP-Verfahren berechneten Gradienten von Zielfunktional und Nebenbedingungen Wert gelegt. Diese werden mit semianalytischen Ableitungen (z. B. des Zielfunktionals), Automatischer Differentiation (der Nebenbedingungen, der Modellantwort und der Modellfunktionen des DAE-Systems) und IND zur Generierung der ersten und zweiten Ableitungen der Lösung des DAE-Systems berechnet. Die sequentielle Vorgehensweise bietet ein effizientes Tool zur Behandlung von Versuchsplanungsproblemen zur Parameterschätzung.

Die Vergleiche mit anderen Integratoren beweisen die Leistungsfähigkeit von DAESOL zur Simulation und Ableitungsgenerierung. Am Beispiel der Phosphin- und Urethan-Reaktion wird gezeigt, daß DAESOL erfolgreich in komplexen Optimierungsproblemen eingesetzt wurde und dadurch eine ganz neue Problemklasse behandelt werden konnte.

8.2 Ausblick

Die Ergebnisse haben gezeigt, daß DAESOL ein leistungfähiges Tool ist, das im Durchschnitt sowohl für die reine Simulation, aber erst recht für die Berechnung erster und zweiter Ableitungen der Lösungstrajektorie, deutlich schneller ist als die anderen Integratoren. Außerdem ist es das einzige Tool, das zweite Ableitungen generiert und Richtungsableitungen direkt berechnet. Es wird im Zusammenhang mit der Parameterschätzung, der Versuchsplanung und der Optimalsteuerung sehr erfolgreich eingesetzt.

Die bisher mit DAESOL behandelten Systeme hatten bis zu 2000 Zustandsvariablen. Die effiziente numerische Behandlung dieser Systeme mit der Dense- bzw. Sparse-Version von DAESOL mit den zugehörigen Lineare-Algebra-Lösern wurde im vorhergehenden Kapitel gezeigt. Zur Behandlung von sehr großen Systemen mit unstrukturierten Jacobi-Matrizen ist die Lösung der Lineare-Algebra-Teilprobleme mit einem iterativen Verfahren (etwa GMRES von Saad und Schultz [SS86] oder die Weiterentwicklung von Saad und Wu [SW96]) vorzuziehen. Aufgrund der modularen Struktur von DAESOL können andere Lineare-Algebra-Löser sehr einfach eingebunden werden.

In der Regel sind die aus der Modellierung von chemischen und verfahrenstechnischen Prozessen resultierenden DAE-Systeme vom Index 0 oder 1. Zur numerischen Behandlung von DAEs von höherem Index sollte zusätzlich eine Projektion der Lösungstrajektorie auf die Invarianten erfolgen, wie es zum Beispiel in den Integratoren von Eich [Eic92, Eic93], Petzold [BCP96] und von Schwerin [vS97] implementiert ist. Bei der Berechnung von Ableitungen der Lösungstrajektorie muß nach den Prinzipien der IND die Projektion mit abgeleitet werden. Die Methoden hierfür wurden erstmals von Schulz et al. [SBS98] untersucht. Von Schwerin [vS97] hat diese Methoden zur Lösung von Anfangswertproblemen und zur Ableitungsgenerierung im Optimierungskontext für mechanische Systeme vom Index 3 implementiert.

Enthalten die DAE-Systeme zustandsabhängige Unstetigkeiten oder Nichtdifferenzierbarkeiten, so kann das von Winckler [Win97] entwickelte Modul, das eine automatische Unste-

tigkeitenbehandlung erlaubt, an DAESOL angekoppelt werden. Aufgrund der kontinuierlichen Lösungsdarstellung kann die Trajektorie für die Schaltpunktbehandlung fehlerkontrolliert und praktisch ohne zusätzlichen Aufwand auch an Zwischenwerten ausgewertet werden.

Alternativ zu den beiden Verfahren zur Ableitungsgenerierung in DAESOL (direkte und iterative Methode) ist die Berechnung über adjungierte Gleichungen (siehe Bock [Boc87]) gerade dann sinnvoll, wenn die Anzahl der abzuleitenden Funktionen (in der Regel Zielfunktional und Nebenbedingungen) gering ist.

Die Programmpakete zur Simulation, Parameterschätzung und Versuchsplanung werden bereits erfolgreich bei der BASF AG eingesetzt.

Literaturverzeichnis

[AB85a] Agarwal, A. K. und M. L. Brisk: *Sequential Experimental Design for Precise Parameter Estimation. 1. Use of Reparameterization.* Ind. Eng. Chem. Process Des. Dev., 24:203–207, 1985.

[AB85b] Agarwal, A. K. und M. L. Brisk: *Sequential Experimental Design for Precise Parameter Estimation. 2. Design Criteria.* Ind. Eng. Chem. Process Des. Dev., 24:207–210, 1985.

[ABB+95] Anderson, E., Z. Bai, Ch. H. Bischof, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov und D. Sorensen: *LAPACK user's guide.* SIAM, Philadelphia, PA, 2. Auflage, 1995.

[ACR81] Ascher, U. M., J. Christiansen und R. Russell: *Collocation software for boundary value ODEs.* ACM Trans. Math. Software, 7:209–222, 1981.

[AD92] Atkinson, A. C. und A. N. Donev: *Optimum Experimental Designs.* Oxford Statistical Science Series, Oxford University Press, 1992.

[Arn91] Arnold, M.: *Applying BDF to quasilinear differential-algebraic equations of index 2: perturbation analysis.* Technischer Bericht, Universität Rostock, 1991.

[Atk96] Atkinson, A. C.: *The Usefulness of Optimum Experimental Design.* Royal Statistical Society, 58(1):59–76, 1996.

[BA87] Bader, G. und U. M. Ascher: *A new basis implementation for a mixed order boundary value ODE solver.* SIAM J. Sci. Comput., 8:483–500, 1987.

[Bar74] Bard, Y.: *Nonlinear Parameter Estimation.* Academic Press, Inc., San Diego, 1974.

[Bär83] Bär, V.: *Ein Kollokations-Verfahren zur numerischen Lösung allgemeiner Mehrpunktrandwertaufgaben mit Schalt- und Sprungbedingungen mit Anwendungen in der optimalen Steuerung und der Parameteridentifizierung.* Diplomarbeit, Universität Bonn, 1983.

[Bau72] Baumgarte, J.: *Stabilization of constraints and integrals of motion in dynamical systems.* Comp. Meth. Appl. Mech. Engng., 1:1–16, 1972.

[Bau94] Bauer, I.: *Numerische Behandlung Differentiell-Algebraischer Gleichungen mit Anwendungen in der Chemie.* Diplomarbeit, Universität Augsburg, 1994.

[BBH89] Brown, P. N., G. D. Byrne und A. C. Hindmarsh: *VODE: A variable-coefficient ODE solver.* SIAM J. Sci. Stat. Comput., 10(5):1038–1051, 1989.

[BBKS99a] Bauer, I., H. G. Bock, S. Körkel und J. P. Schlöder: *Numerical Methods for Initial Value Problems and Derivative Generation for DAE Models with Application to Optimum Experimental Design of Chemical Processes.* In [KMVW99], Seiten 282–289, 1999.

[BBKS99b] Bauer, I., H. G. Bock, S. Körkel und J. P. Schlöder: *Numerical Methods for Optimum Experimental Design in DAE systems*, 1999. Erscheint in Journ. Comp. Appl. Math.

[BBLS99] Bauer, I., H. G. Bock, D. B. Leineweber und J. P. Schlöder: *Direct Multiple Shooting Methods for Control and Optimization of DAE in Chemical Engineering.* In [KMVW99], Seiten 2–18, 1999.

[BBMP90] Bachmann, R., L. Brüll, Th. Mrziglod und U. Pallaske: *On methods for reducing the index of differential algebraic equations.* Comp. chem. Engng., 14:1271–1273, 1990.

[BBS99] Bauer, I., H. G. Bock und J. P. Schlöder: *DAESOL – a BDF-code for the numerical solution of differential algebraic equations.* Internal report, IWR, SFB 359, Universität Heidelberg, 1999.

[BCC+92] Bischof, Ch. H., A. Carle, G. Corliss, A. Griewank und P. Hovland: *ADIFOR Generating derivative codes from Fortran programs.* Scientific Programming, 1:11–29, 1992.

[BCK+98] Bischof, Ch. H., A. Carle, P. M. Khademi, A. Mauer und P. Hovland: *ADIFOR 2.0 User's Guide.* Technical Memorandum No. 192, Mathematics and Computer Science Division, 1998.

[BCP96] Brenan, K. E., S. L. Campbell und L. R. Petzold: *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.* Classics in Applied Mathematics. SIAM, Philadelphia, PA, 1996.

[BDB86] Biegler, L. T., J. J. Damiano und G. E. Blau: *Nonlinear parameter estimation: a case study comparison.* AIChE J., 32:29–45, 1986.

[BES88] Bock, H. G., E. Eich und J. P. Schlöder: *Numerical solution of constrained least squares boundary value problems in differential-algebraic equations.* In:

Strehmel, K. (Herausgeber): *Numerical Treatment of Differential and Integral Equations*. BG Teubner, Leipzig, 1988.

[BFD+97] Bauer, I., F. Finocchi, W. J. Duschl, H.-P. Gail und J. P. Schlöder: *Simulation of chemical reactions and dust destruction in protoplanetary accretion disks*. Astronomy & Astrophys., 317:273–289, 1997.

[BGS88] Brankin, R. W., I. Gladwell und L. F. Shampine: *Starting BDF and Adams codes at optimal order*. J. Comput. Appl. Math., 21(3):357–368, 1988.

[BHB98] Brown, P. N., A. C. Hindmarsh und G. D. Byrne: *VODE : A variable-coefficient ODE solver*, May 15, 1997. Bug fix release November 12, 1998. Erhältlich unter http://www.netlib.org/ode/vode.f.

[BHH78] Box, G. E. P., W. G. Hunter und J. S. Hunter: *Statistics for Experimenters*. Probability and Mathematical Statistics. Wiley Series, 1978.

[BHK+98] Bauer, I., M. Heilig, S. Körkel, A. Kud, A. Mayer und O. Wörz: *Versuchsplanung am Beispiel einer Phosphin- und Urethanreaktion*. In: *Optimale Versuchsplanung für nichtlineare Prozesse*. DECHEMA e.V., 1998.

[BHP98] Brown, P. N., A. C. Hindmarsh und L. R. Petzold: *Consistent initial condition calculation for differential-algebraic systems*. SIAM J. Sci. Comput., 19(5):1495–1512, 1998.

[BJL+93] Bilardello, P., X. Joulia, J. M. LeLann, H. Delmas und B. Koehret: *A general strategy for parameter estimation in differential-algebraic systems*. Comp. chem. Engng., 7:517–525, 1993.

[Bjö96] Björck, Å.: *Numerical Methods For Least Squares Problems*. SIAM, Philadelphia, PA, 1996.

[BKBS98] Bauer, I., S. Körkel, H. G. Bock und J. P. Schlöder: *Optimale Versuchsplanung für dynamische Systeme aus der chemischen Reaktionskinetik*. In: *Optimale Versuchsplanung für nichtlineare Prozesse*. DECHEMA e.V., 1998.

[BL59] Box, G. E. P. und H. L. Lucas: *Design of experiments in non-linear situations*. Biometrika, 46:77–90, 1959.

[Ble86] Bleser, G.: *Eine effiziente Ordnungs– und Schrittweitensteuerung unter Verwendung von Fehlerformeln für variable Gitter und ihre Realisierung in Mehrschrittverfahren vom BDF–Typ*. Diplomarbeit, Universität Bonn, 1986.

[Boc78] Bock, H. G.: *Numerical Solution of Nonlinear Multipoint Boundary Value Problems with Application to Optimal Control*. Z. Angew. Math. Mech., 58:T407–T409, 1978.

[Boc81] Bock, H. G.: *Numerical Treatment of Inverse Problems in Chemical Reaction Kinetics.* In: Ebert, K. H., P. Deuflhard und W. Jäger (Herausgeber): *Modelling of Chemical Reaction Systems*, Band 18 der Reihe *Springer Series in Chemical Physics 18*, Seiten 102–125, Heidelberg, 1981.

[Boc83] Bock, H. G.: *Recent Advances in Parameter Identification Techniques for ODE.* In [DH83], Seiten 95–121, 1983.

[Boc87] Bock, H. G.: *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen.* Bonner Mathematische Schriften 183, 1987.

[Box60] Box, G. E. P.: *Fitting Empirical Data.* Ann. N. Y. Acad. Sci., 86:792–816, 1960.

[BS81] Bock, H. G. und J. P. Schlöder: *Numerical solution of retarded differential equations with state dependent time lags.* Z. Angew. Math. Mech., 61:T269–T271, 1981.

[BS86] Bock, H. G. und J. P. Schlöder: *Recent progress in the development of algorithms and software for large scale parameter estimation problems in chemical reaction systems.* In: Kotobh, P. (Herausgeber): *Automatic Control in Petrol, Petrochemical and Desalination Industries.* IFAC Congress, Pergamon, Oxford, 1986.

[BSSR94] Baltes, M., R. Schneider, C. Sturm und M. Reuss: *Optimal Experimental Design for Parameter Estimation in Unstructured Growth Models.* Biotechnol. Prog., 10:480–488, 1994.

[Bul71] Bulirsch, R.: *Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung.* Technischer Bericht, Carl-Cranz-Gesellschaft, 1971.

[BW88] Bates, D. M. und D. G. Watts: *Nonlinear Regression Analysis And Its Applications.* Wiley Series In Probability And Mathematical Statistics. John Wiley & Sons, Inc., 1988.

[CB87] Cuthrell, J. E. und L. T. Biegler: *On the optimization of differential-algebraic process systems.* AIChE J., 33:1257–1270, 1987.

[CH52] Curtiss, C. F. und J. O. Hirschfelder: *Integration of stiff equations.* Proc. Nat. Acad. Sci., 38:235–243, 1952.

[CL84] Crouzeix, M. und F. J. Lisbona: *The convergence of variable-stepsize, variable-formula, multistep-methods.* SIAM J. Numer. Anal., 21:512–534, 1984.

[CLM87] Calvo, M., F. J. Lisbona und J. Montijano: *On the stability of variable step-size Nordsieck BDF methods.* SIAM J. Numer. Anal., 24:844–854, 1987.

[Cry72] Cryer, C. W.: *On the instability of high order backward-difference multistep methods.* BIT, 12:17–25, 1972.

[CS85] Caracotsios, M. und W. E. Stewart: *Sensitivity analysis of initial value problems with mixed ODEs and algebraic equations.* Comp. chem. Engng., 9:359–365, 1985.

[CS95] Caracotsios, M. und W. E. Stewart: *Sensitivity analysis of initial value problems with mixed PDEs and algebraic equations.* Comp. chem. Engng., 19:1019–1030, 1995.

[Dah56] Dahlquist, G.: *Convergence and stability in the numerical integration of ordinary differential equations.* Math. Scand., 4:33–53, 1956.

[DBMS80] Dongarra, J. J., J. R. Bunch, C. B. Moler und G. W. Stewart: *LINPACK, user's guide.* SIAM, Philadelphia, PA, 2. Auflage, 1980.

[DD95] Davis, T. A. und I. S. Duff: *An unsymmetric-pattern multifrontal method for sparse LU factorization.* SIAM J. Matrix Anal. Appl., 18:140–158, 1995.

[Deu74] Deuflhard, P.: *A Modified Newton Method for the Solution of Ill-conditioned Systems of Nonlinear Equations with Applications to Multiple Shooting.* Numer. Math., 22:289–315, 1974.

[DH66] Draper, N. R. und W. G. Hunter: *Design of experiments for parameter estimation in multiresponse situations.* Biometrika, 53:525, 1966.

[DH83] Deuflhard, P. und E. Hairer (Herausgeber): *Numerical Treatment of Inverse Problems in Differential and Integral Equations*, Band 2, Birkhäuser, Boston, 1983.

[DHZ87] Deuflhard, P., E. Hairer und J. Zugck: *One-step and extrapolation methods for differential-algebraic systems.* Numer. Math., 51:501–516, 1987.

[Die97] Dieses, A. E.: *Numerische Verfahren zur Diskriminierung nichtlinearer Modelle für dynamische chemische Prozesse.* Diplomarbeit, IWR, Universität Heidelberg, 1997.

[DP96] Draper, N. R. und F. Pukelsheim: *An overview of design of experiments.* Statistical Papers, 37:1–32, 1996.

[DR96] Duff, I. S. und J. K. Reid: *The design of MA48: A code for the direct solution of sparse unsymmetric linear systems of equations.* ACM Trans. Math. Softw., 22:187–226, 1996.

[DRAD94] Dovíc, V. G., A. P. Reverberi und L. Acevedo-Duarte: *New Procedure for Optimal Design of Sequential Experiments in Kinetic Models*. Ind. Eng. Chem. Res., 33:62–68, 1994.

[DS83] Dennis, J. E. und R. B. Schnabel: *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice Hall, Englewood Cliffs, 1983.

[DSBR99] Dieses, A. E., J. P. Schlöder, H. G. Bock und O. Richter: *Parameter Estimation for Nonlinear Transport and Degradation Processes of Xenobiotica in Soil*. In [KMVW99], Seiten 290–297, 1999.

[Dun84] Dunker, A. M.: *The decoupled direct method for calculating sensitivity coefficients in chemical kinetics*. J. Chem. Phys., 81:2385–2393, 1984.

[EH74] Emig, G. und L. H. Hosten: *On the reliability of parameter estimates in a set of simultaneous nonlinear differential equations*. Chem. Eng. Sci., 29:475–483, 1974.

[EHL75] Enright, W. H., E. T. Hull und B. Lindberg: *Comparing numerical methods for stiff systems of O.D.E.s*. BIT, 15:10–48, 1975.

[Eic87] Eich, E.: *Numerische Behandlung semi-expliziter differentiell-algebraischer Gleichungssysteme vom Index 1 mit BDF-Verfahren*. Diplomarbeit, Universität Bonn, 1987.

[Eic92] Eich, E.: *Projizierende Mehrschrittverfahren zur numerischen Lösung von Bewegungsgleichungen technischer Mehrkörpersysteme mit Zwangsbedingungen und Unstetigkeiten*. Dissertation, Universität Augsburg 1991, erschienen als Fortschr.-Ber. VDI Reihe 18 Nr. 109. VDI-Verlag, Düsseldorf 1992.

[Eic93] Eich, E.: *Convergence results for a coordinate projection method applied to mechanical systems with algebraic constraints*. SIAM J. Numer. Anal., 30:1467–1482, 1993.

[EN99] Ehrig, R. und U. Nowak: *LIMEX Version 4.1A*, July 19, 1999 1999. Erhältlich unter http://www.zib.de/nowak/limex4.html.

[EW95] Edsberg, L. und P.-A. Wedin: *Numerical tools for parameter estimation in ODE-systems*. Optim. Meth. Softw., 16:193–217, 1995.

[FBGS96] Finocchi, F., I. Bauer, H.-P. Gail und J. P. Schlöder: *Chemical reactions and dust destruction in protoplanetary accretion disks*. In: Warnatz, J. und F. Behrendt (Herausgeber): *Proc. of Int. Workshop on Modelling of Chemical Reactions, July 24-26, 1996*, IWR, Heidelberg, Germany, 1996.

[FCPD90] Farhat, S., M. Czernicki, L. Pibouleau und S. Domenech: *Optimization of multiple-fraction batch distillation by nonlinear programming*. AIChE J., 36:1349–1360, 1990.

[Fed72] Fedorov, V. V.: *Theory of Optimal Experiments*. Probability and Mathematical Statistics. Academic Press, Inc. (London) LTD., 1972.

[FTB97] Feehery, W. F., J. E. Tolsma und P. I. Barton: *Efficient sensitivity analysis of large-scale differential-algebraic systems*. Appl. Numer. Math., 25:41–54, 1997.

[Gal97] Gallitzendörfer, J. V.: *Parallele Algorithmen für Optimierungsrandwertprobleme*. Dissertation, Universität Heidelberg 1996, erschienen als Fortschr.-Ber. VDI Reihe 10 Nr. 514. VDI-Verlag, Düsseldorf, 1997.

[GB94] Gallitzendörfer, J. V. und H. G. Bock: *Parallel Algorithms for Optimization Boundary Value Problems in DAE*. In: H. Langendörfer (Herausgeber): *Praxisorientierte Parallelverarbeitung*. Hanser, München, 1994.

[Gea71] Gear, C. W.: *Simultaneous numerical solution of differential–algebraic equations*. IEEE Trans. Circuit Theory, CT-18(1):89–95, 1971.

[Gea80] Gear, C. W.: *Runge-Kutta starters for multistep methods*. ACM Trans. Math. Softw., 6:263–279, 1980.

[Gea88] Gear, C. W.: *Differential-algebraic equation index transformations*. SIAM J. Sci. Stat. Comput., 9:39–47, 1988.

[GH93] Gander, W. und J. Hrebicek: *Solving problems in scientific computing using MAPLE and MATLAB*. Springer, Berlin, 1993.

[GMS97a] Gill, Ph. E., W. Murray und M. A. Saunders: *SNOPT: an SQP Algorithm for Large-Scale Constrained Optimization*. Technischer Bericht, NA 97-2, Department of Mathematics, University of California, San Diego, and Report SOL 97-3, Dept. of EESOR, Stanford University, 1997.

[GMS97b] Gill, Ph. E., W. Murray und M. A. Saunders: *User's Guide for SNOPT 5.3: a Fortran Package for Large-Scale Nonlinear Programming*. Technischer Bericht, NA 97-5, Department of Mathematics, University of California, San Diego, 1997.

[Gri77] Grigorieff, R. D.: *Numerik gewöhnlicher Differentialgleichungen 2*. Teubner Studienbücher Mathematik. B. G. Teubner, Stuttgart, 1977.

[Gri83] Grigorieff, R. D.: *Stability of Multistep–Methods on Variable Grids*. Numer. Math., 42:359–377, 1983.

[GT74] Gear, C. W. und K. W. Tu: *The effect of variable mesh size on the stability of multistep methods*. SIAM J. Numer. Anal., 11:1025–1043, 1974.

[GV83] Gear, C. W. und T. Vu: *Smooth Numerical Solution of Ordinary Differential Equations*. In [DH83], Seiten 2–12, 1983.

[GW74] Gear, C. W. und D. S. Watanabe: *Stability and convergence of variable order multistep methods*. SIAM J. Numer. Anal., 11:1044–1058, 1974.

[Hil96] Hilf, K.-D.: *Optimale Versuchsplanung zur dynamischen Roboterkalibrierung*. Dissertation, Universität Heidelberg 1996, erschienen als Fortschr.-Ber. VDI Reihe 8 Nr. 590. VDI-Verlag, Düsseldorf, 1996.

[Hin80] Hindmarsh, A. C.: *LSODE and LSODI, two new initial value ordinary differential equation solvers*. ACM-SIGNUM Newsletters, 15:10–11, 1980.

[HLR89] Hairer, E., C. Lubich und M. Roche: *The Numerical Solution of Differential–Algebraic Systems by Runge-Kutta Methods*, Band 1409 der Reihe *Lecture Notes in Mathematics*. Springer Verlag, New York, 1989.

[HNW93] Hairer, E., S. P. Nørsett und G. Wanner: *Solving Ordinary Differential Equations I: Nonstiff problems*. Springer Berlin Heidelberg, 2. rev. Auflage, 1993.

[Hub96] Huber, P. J.: *Robust statistics, data analysis, and computer intensive methods*. Lecture Notes in Statistics; 109. Springer, 1996.

[Hug97] Hugo, P.: *Thermokinetische Auswertung von adiabatischen Semibatch-Messungen*. http://www.tu-berlin.de/~itc/hugo/asbr/asbr.html, TU Berlin, 1997.

[HW88] Hairer, E. und G. Wanner: *RADAU5 - an implicit Runge-Kutta code*. Report, Université de Genève, Dept. de mathématiques, Genève, 1988.

[HW96a] Hairer, E. und G. Wanner: *RADAU5*, July 9 1996. Erhältlich unter ftp://ftp.unige.ch/pub/doc/math/stiff/radau5.f.

[HW96b] Hairer, E. und G. Wanner: *Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems*. Springer Berlin Heidelberg, 2. rev. Auflage, 1996.

[HW98] Hairer, E. und G. Wanner: *RADAU*, September 18 1998. Erhältlich unter ftp://ftp.unige.ch/pub/doc/math/stiff/radau.f.

[KBBS99] Körkel, S., I. Bauer, H. G. Bock und J. P. Schlöder: *A sequential approach for nonlinear optimum experimental design in DAE systems*. In [KMVW99], Seiten 338–345, 1999.

[Kie61] Kiefer, J. C.: *Optimum Designs in Regression Problems II*. Ann. Math. Stat., 32:298–325, 1961.

[Kie75] Kiefer, J. C.: *Optimal design: variation in structure and performance under change of criterion*. Biometrika, 62:277–288, 1975.

[KMVW99] Keil, F., W. Mackens, H. Voss und J. Werther (Herausgeber): *Scientific Computing in Chemical Engineering II*, Band 2: Simulation, Image Processing, Optimization, and Control. Springer-Verlag, Berlin, Heidelberg, 1999.

[Kof96] Kofler, M.: *Maple V Release 4: Einführung und Leitfaden für den Praktiker.* Addison-Wesley, Bonn, 1996.

[KW59] Kiefer, J. C. und J. Wolfowitz: *Optimum Designs in Regression Problems.* Ann. Math. Stat., 30:271–294, 1959.

[LB89] Logsdon, J. S. und L. T. Biegler: *Accurate Solution of Differential-Algebraic Optimization Problems.* Ind. Eng. Chem. Res., 28:1628–1639, 1989.

[LBS92] Lohmann, Th. W., H. G. Bock und J. P. Schlöder: *Numerical Methods for Parameter Estimation and Optimal Experiment Design in Chemical Reaction Systems.* Ind. Eng. Chem. Res., 31:54–57, 1992.

[Lei99] Leineweber, D. B.: *Efficient reduced SQP methods for the optimization of chemical processes described by large sparse DAE models.* Dissertation, Universität Heidelberg, 1999.

[Loh88] Lohmann, Th. W.: *Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen, Modellbildung durch Analyse der Kovarianzmatrix der Parameterschätzung.* Diplomarbeit, Universität Bonn, 1988.

[Loh93] Lohmann, Th. W.: *Ein numerisches Verfahren zur Berechnung optimaler Versuchspläne für beschränkte Parameteridentifizierungsprobleme.* Reihe Informatik. Verlag Shaker, Aachen, 1993.

[LP86] Lötstedt, P. und L. R. Petzold: *Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints I: Convergence Results for Backward Differentiation Formulas.* Math. Comp., C46:491–516, 1986.

[LP99a] Li, S. und L. R. Petzold: *Design of New DASPK for Sensitivity Analysis.* Technischer Bericht, UCSB, Department of Computer Science, 1999.

[LP99b] Li, S. und L. R. Petzold: *Large Scale Differential Algebraic Equation Solver DASPK*, May, 4 1999. Erhältlich unter http://www.engineering.ucsb.edu/~cse/daspk.tar.

[LS] Lioen, W. M. und J. J. B. Swart: *Test Set for Initial Value Problem Solvers.* Release 2.1, September 1999. Erhältlich unter http://www.cwi.nl/cwi/projects/IVPtestset/.

[MP96] Maly, T. und L. R. Petzold: *Numerical methods and software for sensitivity analysis of differential-algebraic systems.* Appl. Numer. Math., 20:57–79, 1996.

[MS93] Mattsson, S. und G. Söderlind: *Index reduction in differential-algebraic equations using dummy derivatives*. SIAM J. Sci. Comput., 14:677–692, 1993.

[MW93] Moré, J. J. und S. J. Wright: *Optimization Software Guide*, Band 14 der Reihe *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, 1993.

[NAG91] *The Numerical Algorithms Group Ltd.; Oxford, UK*. The NAG FORTRAN Library Manual, Mark 15, 1991.

[NM64] Nelder, J. A. und R. A. Mead: *A simplex method for function minimisation*. Comput. J., 7:308–313, 1964.

[Nor62] Nordsieck, A.: *On numerical integration of ordinary differential equations*. Math. Comput., 16:22–49, 1962.

[OR66] Ortega, J. M. und W. C. Rheinboldt: *On discretization and differentiation of operators with application to Newton's method*. SIAM J. Numer. Anal., 3:143–156, 1966.

[OTH92] Oinas, P., I. Turunen und H. Haario: *Experimental Design With Steady-State And Dynamic Models Of Multiphase Reactors*. Chemical Engineering Science, 47(13/14):3689–3696, 1992.

[Pan88] Pantelides, C. C.: *The consistent initialization of differential–algebraic systems*. SIAM J. Sci. Stat. Comput., 9(2):213–231, 1988.

[Páz86] Pázman, A.: *Foundations of Optimum Experimental Design*. Reidel, Dordrecht, 1986.

[Pet82a] Petzold, L. R.: *A Description of* DASSL*: A Differential–Algebraic System Solver*. In: *Proc. 10th IMACS World Congress, August 8-13 Montreal 1982*, 1982.

[Pet82b] Petzold, L. R.: *Differential/algebraic equations are not ODEs*. SIAM J. Sci. Stat. Comput., 3(2):367–386, 1982.

[Pet91] Petzold, L. R.: *DASSL: A Differential/Algebraic System Solver*, June 24 1991. Erhältlich unter http://www.netlib.org/ode/ddassl.f und unter http://www.engineering.ucsb.edu/~cse/ddassl.tar.gz.

[PL86] Petzold, L. R. und P. Lötstedt: *Numerical Solution of Nonlinear Differential Equations with Algebraic Constraints II: Practical Implementations*. SIAM J. Sci. Stat. Comput., 7:720–733, 1986.

[PRM97] Petzold, L. R., Y. Ren und T. Maly: *Regularization of higher-index differential-algebraic equations with rank-deficient constraints*. SIAM J. Sci. Comput., 18:753–774, 1997.

[PSV94] Pantelides, C. C., R. W. H. Sargent und V. S. Vassiliadis: *Optimal control of multistage systems described by high-index differential-algebraic equations.* In: Bulirsch, R. und D. Kraft (Herausgeber): *Computational optimal control. Proceedings of the 9th IFAC workshop on control applications of optimization.* Birkhäuser, Basel, 1994.

[Puk93] Pukelsheim, F.: *Optimal Design Of Experiments.* Wiley Series In Probability And Mathematical Statistics. John Wiley & Sons, Inc., 1993.

[QNG80] Qureshi, Z. H., T. S. Ng und G. C. Goodwin: *Optimum experimental design for identification of distributed parameter systems.* Int. J. Control, 31(1):21–29, 1980.

[Ras90] Rasch, D.: *Optimum Experimental Design In Nonlinear Regression.* Commun. Stat. Theor. Meth., 19(12):4789–4806, 1990.

[Rau97] Raum, A.: *Numerische Methoden für die Optimale Versuchsplanung bei nichtlinearen Problemen.* Diplomarbeit, IWR, Universität Heidelberg, 1997.

[RBB+77] Reilly, P. M., R. Bajramovic, G. E. Blau, D. R. Branson und M. W. Sauerhoff: *Guidelines for the Optimal Design of Experiments to Estimate Parameters in First Order Kinetic Models.* The Canadian Journal of Chemical Engineering, 55:614, 1977.

[RH95] Rudolph, P. E. und G. Herrendörfer: *Optimal Experimental Design and Accuracy of Parameter Estimation for Nonlinear Regression Models Used in Long-term Selection.* Biom. J., 37(2):183–190, 1995.

[Rhe84] Rheinboldt, W. C.: *Differential–Algebraic Systems as Differential Equations on Manifolds.* Math. of Comp., 43:473–482, 1984.

[RMA87] Renfro, J. G., A. M. Morshedi und O. A. Asbjornsen: *Simultaneous Optimization and Solution of Systems Described by Differential/Algebraic Equations.* Comp. chem. Engng., 11:503–517, 1987.

[RR94a] Rabier, P. J. und W. C. Rheinboldt: *On Impasse Points of Quasilinear Differential–Algebraic Equations.* J. Math. Anal. Appl., 181(2):429–454, 1994.

[RR94b] Rabier, P. J. und W. C. Rheinboldt: *On the Computation of Impasse Points of Quasilinear Differential–Algebraic Equations.* Math. Comp., 62(101):133–154, 1994.

[Rüc99] Rücker, G.: *Automatisches Differenzieren mit Anwendung in der Optimierung bei chemischen Reaktionssystemen.* Diplomarbeit, IWR, Universität Heidelberg, 1999.

[SB73] Stoer, J. und R. Bulirsch: *Numerische Mathematik 2*. Springer Verlag, Berlin, 1973.

[SB83] Schlöder, J. P. und H. G. Bock: *Identification of Rate Constants in Bistable Chemical Reactions*. In [DH83], Seiten 27–47, 1983.

[SB89] Shampine, L. F. und P. Bogacki: *The effect of changing the stepsize in linear multistep codes*. SIAM J. Sci. Stat. Comput., 10(5):1010–1023, 1989.

[SBS98] Schulz, V. H., H. G. Bock und M. C. Steinbach: *Exploiting Invariants in the Numerical Solution of Multipoint Boundary Value Problems in DAE*. SIAM J. Sci. Comp., 19:440–467, 1998.

[Sch88] Schlöder, J. P.: *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung*. Bonner Mathematische Schriften 187, 1988.

[Sch90] Schulz, V. H.: *Ein effizientes Kollokationsverfahren zur numerischen Behandlung von Mehrpunktrandwertaufgaben in der Parameteridentifizierung und Optimalen Steuerung*. Diplomarbeit, Universität Augsburg, 1990.

[Sch99a] Schittkowski, K.: *EASY-FIT: A software system for data fitting in dynamic systems*. Technischer Bericht, Fakultät für Mathematik, Universität Bayreuth, 1999. submitted for publication.

[Sch99b] Schittkowski, K.: *PDEFIT: A FORTRAN code for data fitting in partial differential equations*. Optim. Meth. Softw., 10:539–582, 1999.

[SH80] Sherman, A. H. und A. C. Hindmarsh: *GEARS: A package for the solution of sparse, stiff ordinary differential equations*. In: *Electric power problems: the mathematical challenge*, Proc. Conf., Seiten 190–200, Seattle/Wash., 1980.

[SH99] Støren, S. und T. Hertzberg: *Obtaining sensitivity information in dynamic optimization problems solved by the sequential approach*. Comp. chem. Engng., 23:807–819, 1999.

[SS78] Sargent, R. W. H. und G. R. Sullivan: *The development of an efficient optimal control package*. In: *Optim. Techn., Proc. IFIP Conf.*, Band 2 der Reihe *Lect. Notes Control Inf. Sci. 7*, Würzburg, 1978.

[SS86] Saad, Y. und M. H. Schultz: *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*. SIAM J. Sci. Stat. Comput., 7:856–869, 1986.

[Sto71] Stoer, J.: *On the numerical solution of constrained least-squares problems*. SIAM J. Numer. Anal., 8:382–411, 1971.

[Sto97] Stortelder, W. J. H.: *Parameter Estimation in Nonlinear Dynamical Systems.* Dissertation, CWI, Amsterdam, 1997.

[SW89] Seber, G. A. F. und C. J. Wild: *Nonlinear Regression.* John Wiley & Sons, Inc., 1989.

[SW96] Saad, Y. und K. Wu: *DQGMRES: a direct quasi-minimal residual algorithm based on incomplete orthogonalization.* Numer. Linear Algebra Appl., 3:329–343, 1996.

[SZ90] Shampine, L. F. und W. Zhang: *Rate of Convergence of Multistep Codes Started by Variation of Order and Stepsize.* SIAM J. Numer. Anal., 27:1506–1518, 1990.

[Tis95] Tischendorf, C.: *Feasibility and stability behaviour of the BDF applied to index-2 differential algebraic equations.* Z. Angew. Math. Mech., 75(12):927–946, 1995.

[VB90] Vasantharajan, S. und L. T. Biegler: *Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria.* Comp. chem. Engng., 14:1083–1100, 1990.

[vS97] Schwerin, R. von: *Numerical Methods, Algorithms, and Software for Higher Index Nonlinear Differential-Algebraic Equations in Multibody System Simulation.* Dissertation, Universität Heidelberg, 1997.

[vS98] Schwerin, M. von: *Numerische Methoden zur Schätzung von Reaktionsgeschwindigkeiten bei der katalytischen Methankonversion und Optimierung von Essigsäure– und Methanprozessen.* Dissertation, Universität Heidelberg, 1998.

[vSB95] Schwerin, R. von und H. G. Bock: *A Runge-Kutta-Starter for a Multistep-Method for Differential-Algebraic Systems with Discontinuous Effects.* APNUM, 18:337–350, 1995.

[Wal80] Walter, W.: *Gewöhnliche Differentialgleichungen.* Heidelberger Taschenbücher. Springer Verlag, Berlin, 3. Auflage, 1980.

[Win97] Winckler, M.: *Semiautomatic discontinuity treatment in FORTRAN77-coded ODE models.* In: *Scientific Computation, Modelling and Applied Mathematics*, Berlin, 1997. 15th IMACS World Congress 1997.

[ZBGS96] Zieße, M. W., H. G. Bock, J. V. Gallitzendörfer und J. P. Schlöder: *Parameter estimation in multispecies transport reaction systems using parallel algorithms.* In: Gottlieb, J. und P. DuChateau (Herausgeber): *Parameter Identification and Inverse Problems in Hydrology, Geology and Ecology*, Seiten 273–282. Kluwer Academic Publishers, 1996.